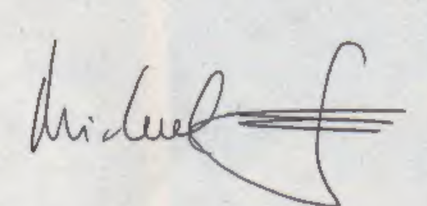
HAPPY-COMPUTER SONDERHEFT

SONDERHEFT 3/1986

DM 14,-

68000er

Markt&Technik



Eine neue Generation von Computern

68000'er – Das Magazin der neuen Computer-Generation. Das ist der Name unseres neuesten Sonderheftes von Happy-Computer. Warum haben wir es 68000'er genannt und den Titel neu konzipiert? Weil wir in dieser Sonderausgabe über einen neuen Computertyp schreiben, der nicht nur einen schnellen und leistungsfähigen Prozessor besitzt, nämlich den 68000, sondern weil dieser Typ von Computer neben viel Speicherplatz und einer Reihe von sehr beeindruckenden Leistungsmerkmalen eine neue Art von Benutzeroberfläche besitzt.

Diese neue Art der Benutzeroberfläche macht es dem Benutzer sehr leicht, mit dem Computer in Verbindung zu treten. Lange, komplizierte und teilweise recht schwierig zu merkende Kommandos gehören der Vergangenheit an. Statt dessen wird ein Befehl symbolisch durch ein leicht erkennbares Zeichen, ein sogenanntes Icon dargestellt. Und dann ist alles ganz leicht: Einfach auf das zeigen, was man haben will – und schon macht der Computer bereitwillig das, was man von ihm verlangt hat. Apropos zeigen, das geht auch ganz einfach: Mit einem kleinen Gerät, der Maus.

So neu ist die Geschichte dieser Benutzeroberfläche übrigens gar nicht. Vor über zehn Jahren hat Rank Xerox bereits diese Art der Benutzeroberfläche vorgestellt. Damals hieß sie noch Smalltalk. Erst mit preiswerten, sehr leistungsfähigen Prozessoren, wie dem 68000 und dem Preisverfall der Speicherbausteine, konnte man diese Form der Benutzeroberfläche zu günstigen Preisen für alle anbieten. Den ersten Schritt machte Apple mit der Lisa und dem Macintosh.

Richtig interessant wurde es allerdings erst, als sich Atari entschloß, einen Computer auf den Markt zu bringen, der mit dieser Art der Benutzeroberfäche arbeitet, einen leistungsfähigen 68000-Prozessor besitzt, ein Megabyte RAM, Farbgrafik und noch vieles mehr. Sensationell war dabei weniger die Technik – die war bekannt. Der Preis des Atari ST sorgte für die Schlagzeilen und dafür, daß diese Art von Computer jetzt plötzlich für jeden erschwinglich war. Ein Computertraum wurde für viele Wirklichkeit.

Bei uns gerade auf dem Markt ist jetzt ein weiterer Vertreter dieses Computertyps – der Amiga von Commodore. Wenngleich vom Preis und von der Leistungsfähigkeit her gesehen etwas anders oder zumindest anders vermarktet als der Atari ST, gibt es nun – mit Apple – insgesamt drei gewichtige Anbieter dieser neuen Computer. Gleich sind die Maschinen dieser drei Hersteller nicht – weder im Preis noch in der Leistungsfähigkeit noch in Ihren von den Marketingabteilungen der Hersteller »geplanten« Anwendungsgebieten.

Aber egal, wohin die Hersteller Ihre Computer verkaufen wollen – wir wollen Ihnen, unseren Lesern erklären, was die neuen Maschinen können, wie man mit ihnen umgeht, was das Faszinierende an ihnen ist und wie man sie programmiert.

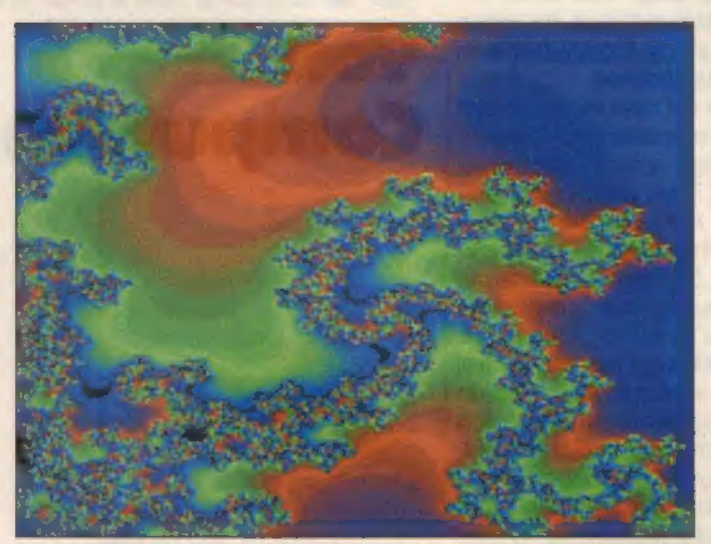
Einen großen Raum dieses Sonderheftes nimmt der Atari ST ein. Warum? Weil er bei uns in Deutschland am meisten verbreitet ist. Weil er ein typischer und dazu noch äußerst preiswerter Vertreter dieses neuen Computertyps ist. Weil man an ihm genausogut wie an einem teureren Computer zeigen kann, wie man ihn programmiert. Und weil unsere Leser viele Informationen über diesen, Ihren Computer haben möchten.

Sicherlich läßt sich an einem Sonderheft, das sich mit dieser Klasse von Computern beschäftigt, vieles verbessern. Helfen Sie uns dabei! Schreiben Sie, was Ihnen gefallen hat, was Ihnen nicht gefallen hat, was fehlt oder was wir das nächste Mal besser machen können!

Michael Scharfenberger, Chefredakteur

Inhalt

Lange Zeit geisterte der Amiga von Commodore nur als Gerücht durch die Computerwelt. Wahre Wunder wurden von ihm erwartet. Jetzt ist er endlich da! Seine Fähigkeiten stehen den Erwartungen kaum nach. Besonders Grafik und Farben sind seine unübersehbaren Stärken. **42**



Die Mandelbrot-Menge ist zwar nur ein abstraktes mathematisches Zahlen-Gebilde. Dennoch enthält es in sich ein ganzes Universum aus Chaos und Schönheit. Sie finden ein Apfelmännchen-Programm für den ST und einen Beitrag zur Theorie der Mandelbrot-Menge. **126**

Benutzerführung

Atari ST, Macintosh, Amiga & Co.

Die besten Drucker, Monitore, Massenspeicher

Statements

Software

Faszinierende Grafik, Spiele und Musik

Einführung: So programmiert man in C-Assembler und GEM

Tips & Tricks für Atari ST

Vermischtes

Komfortable grafische Benutzeroberflächen im Bereich der Personal Computer boten als erste die Apple Computer Lisa und Macintosh. Mit GEM auf dem Atari ST finden sie jetzt auch Eingang in die Klasse unter 3000 Mark. Was nützen sie? 6

Was niemand für möglich gehalten hat, ist Realität geworden: Ataris ST-Modelle werden zur Volks-Vax der neuen Computergeneration – unglaublich preiswert und dennoch technisch vom Feinsten. Atari selbst aber wird zum Trendsetter. 37

Grafik-Power wie einst große Workstations bieten heute Computer der 68000-Generation für wenig Geld. Kein Wunder, daß sich völlig neue Perspektiven für Programmierer auftun. Vor allem Spielprogramme werden davon kräftig profitieren. 106

Mediencomputer oder was ist eine Benutzeroberfläche?

Wie muß eine moderne Benutzeroberfläche gestaltet sein? Sie muß intuitiv sein. Das bedeutet, daß der Mensch vor dem Computer sitzt und tut, was ihm in den Sinn kommt und naheliegend erscheint. Und damit erreicht er genau, was er will.

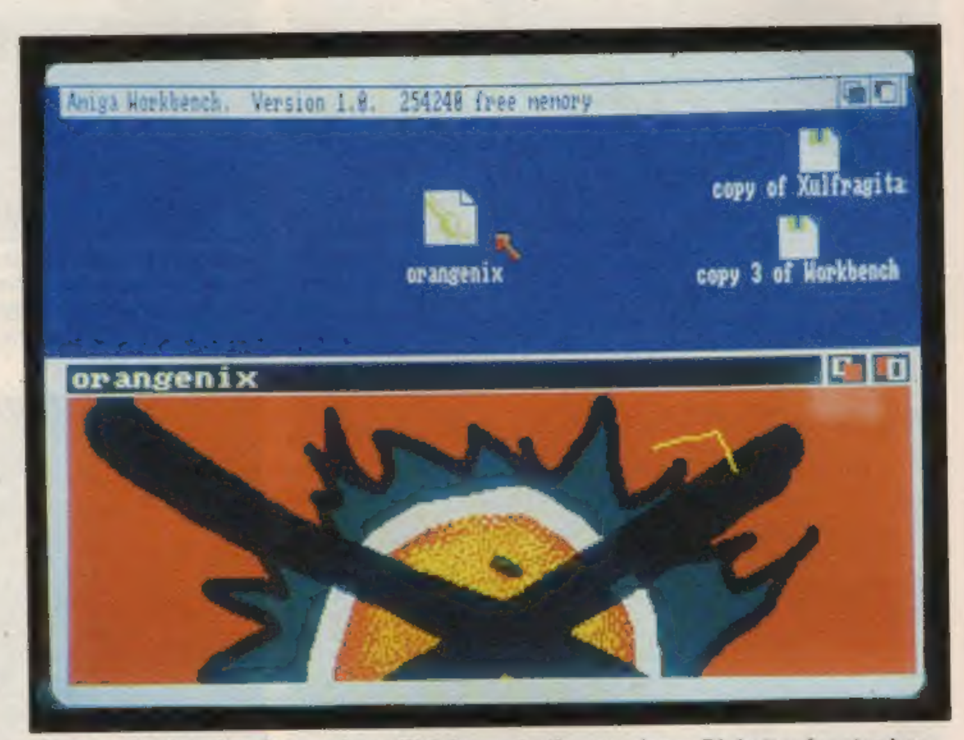
Viele Versuche sind seit Apples Lisa gemacht worden, die grafische Computerumgebung mit Fenstern, Symbolen (Pictogrammen oder Icons) und Maussteuerung auf andere Computer zu übertragen – erinnern wir uns nur an VisiOn. Später kam GEM, dazwischen eine Fülle von Programmen, die mehr oder weniger deutlich von den Konzepten der Lisa von Apple beziehungsweise von Xerox' Star inspiriert waren. Ein modernes Produkt ist zum Beispiel Microsoft Windows.

Doch erst jetzt, zu einem Zehntel des Preises, mit dem Apple die Revolution versuchte, scheint sie zu gelingen: 3000 Mark für einen Computer, der – wie ein Freund schwärmt – außer, daß der Bildschirm kleiner ist, eine Lisp-Maschine in seinem Bedienungskomfort noch übertrifft.

Gedenken wir der Lisa als der Mutter der Mediencomputer für jedermann. Das Wesen der Benutzeroberfläche dieser Computer liegt darin, daß sie dem Menschen eine magische Welt darbieten. In dieser magischen Welt gewinnen Ideen ein surreales Eigenleben: Man kann grafisch dargestellte Gegenstände (Objekte) auf dem Bildschirm anfassen, sie verschieben, verschwinden lassen, vervielfachen. Wenn es sich um Ordner handelt, kann man in sie hineinschauen. Wenn es Zeichnungen sind, kann man die einzelnen Bestandteile größer oder kleiner machen, sie verschieben, mit gemusterten Lacken anmalen, Duplikate herstellen, sie breit oder hoch zerren.

Ehe wir uns in das kleinliche Abwägen verstricken, ob denn nun ein Atari wirklich bietet, was einst Lisa versprach, und was denn mit Amiga ist, doch ein Versuch, zu erklären.

Es kann nur ein Versuch sein. Was wir hier mit stehenden Bildern und tönernen Worten zu fassen suchen, erschließt sich nur, wenn man, wie Ungruen sagen würde, statt eines Kinobesuchs einen Bummel durch die Computerläden macht. Wenn man sich einen Macintosh, Atari oder einen Amiga vorführen läßt – was heißt vorführen, man läßt ihn sich zeigen und probiert ihn dann selber aus. Das sind Computer zum Anfassen, nicht zum Fachsimpeln. Im Gegensatz zu anderen Computern kann man sie sofort bedienen (auch wenn man eine Weile braucht, um ihre ganzen Möglichkeiten zu erkennen).



»Orangenix« ist eine Anwendung (Zeichnung), die von einem Diskettenfenster heruntergenommen und auf dem Schreibtisch abgelegt wurde. Unten sieht man das geöffnete, aber nach unten weggeschobene Arbeitsfenster.

Computer sind intelligente Maschinen – besser gesagt, Maschinen, die intelligente Konzepte oder Programme ausführen können. Programme sind die Seele und der Geist des Computers – oder auch, Programme sind Spiele. Spiele aber haben mit Sehen, Anfassen, Handeln zu tun. Spiele sind motivierend, einleuchtend, entspannend. Und genau das alles sind die neuen Computer – sehr im Gegensatz zu den alten. Schauen – Wollen – Tun, das ist die Losung. Vergessen sind dicke Handbücher, ist das Kopfzerbrechen über Fachausdrücke, das Memorieren abstruser Geheimbefehle. Es ist nicht mehr interessant, daß es Files und Directories gibt, interessant ist nur noch, was wir mit dem Computer tun wollen – ein Bild malen, einen Brief schreiben, etwas ausrechnen.

Das einfachste wäre, wenn wir dem Computer sagen könnten, was wir von ihm wollen. Aber damit ist er – heute noch – überfordert. Die Analyse von Sprache gehört zu den noch nicht bewältigten Problemen der Computerwissenschaft. Doch anders herum funktioniert es wunderbar: Der Computer zeigt uns, was er kann – und wir zeigen ihm, was wir wollen. Grafik heißt das eine Zauberwort, Objekte das andere.

Die Computergrafik hat in den letzten Jahren erstaunliche Fortschritte gemacht. Davon zeugen nicht nur Filme und Werbespots, sondern auch die Fähigkeiten moderner populärer Computer. Mit populären Computern möchte ich die Systeme bezeichnen, von denen man einfach nicht mehr sagen kann, ob es »Heimcomputer« (sprich Spielzeuge) oder »Bürocomputer« (also ernsthafte und nüchterne Arbeitsmittel) sind. Die populären Computer oder Mediencomputer sind einfach beides zugleich – und sie kosten nicht mehr als eine gute Schreibmaschine oder vor ein paar Jahren ein Farbfernseher. Da sie außerdem viel einfacher zu bedienen sind als ein moderner Fernsehapparat, einen ähnlichen Unterhaltungswert haben wie Fernseher und Spielautomat zusammen, haben sie alle Merkmale von Kon-

sumartikeln. Das konnte man bisher von Computern nicht gerade sagen – die waren immer noch etwas für Hacker und Freaks.

Es kommt aber noch etwas hinzu, was ich an anderer Stelle zu begründen suche: Während Apples Lisa und teilweise auch der Mac aus verschiedenen Gründen für die Hacker und Freaks weniger geeignet waren als die eigentlichen Hackercomputer, kommen die neuen Computer den Bedürfnissen beider Gruppen gleichermaßen und ohne Einschränkungen entgegen.

Imagerie: Die Magie auf dem Bildschirm

Die Mediencomputer sind magische Computer. Da sieht man Knöpfe und Schalter auf dem Bildschirm, die man mit Hilfe eines ferngesteuerten Zeigers (Mauscursor) betätigen kann. Möchte man einen Textabsatz in einer anderen Schrift darstellen, fährt man flüchtig mit dem Zeiger darüber – der Absatz erscheint markiert, anschließend läßt man ein Pull-Down-Menü, gewissermaßen eine Jalousie mit Schriftarten oder Mustern zum Aussuchen, herunterspringen und wählt die gewünschte Schrift aus. Und wenn man sich doch einmal in der Bedienung unsicher ist, dann schiebt man das Fenster, in dem man gerade schreibt, einfach beiseite, und schaut sich in einem anderen Fenster das im Computer gespeicherte Handbuch an. Falls nicht schon ein Help-Menü, eine Hilfsjalousie, ausreicht.

Möchte man Sonderzeichen ausländischer Schriften (ç, Ç, é, â, ñ, Ñ) oder wissenschaftliche Sonderzeichen eingeben, die nicht auf einer normalen Tastatur verfügbar sind, so wählt man diese Zeichen (mit Hilfe des Mauszeigers) einfach auf einer Zeichentafel, die ebenfalls im Hintergrund bereitsteht. Neben Fettdruck, unterstrichenem Druck, Kursivschrift und beliebigen Kombinationen dieser Hervorhebungen, erlauben die hochauflösenden Grafikbildschirme in der Regel auch tiefgestellte und hochgestellte Schrift.

Am rechten Rand des Fensters zeigt ein proportionaler Aufzug an, wieviel vom Text auf dem Bildschirm steht und an welcher Position im gesamten Text sich der sichtbare Ausschnitt befindet. Möchte man eine andere Textstelle sehen, so faßt man den Aufzug einfach an und schiebt ihn an die gewünschte Stelle – kurz darauf erscheint der neue Textausschnitt. Das kann man auch mit verschiedenen Texten, die in mehreren Fenstern zu sehen sind, gleichzeitig tun.

Manche der Programme erlauben es sogar schon, Text und Grafik gleichzeitig darzustellen.

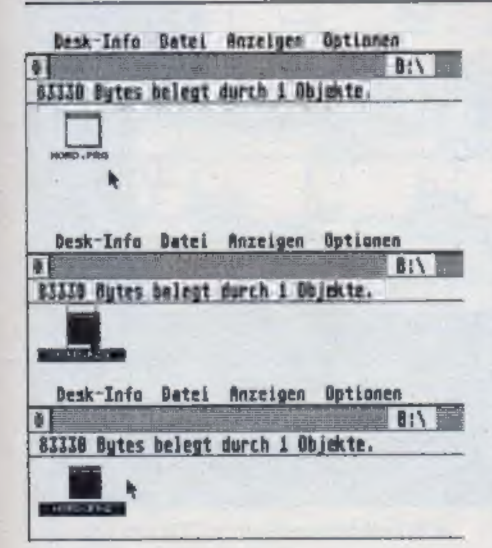
Mit den zugehörigen Zeichenprogrammen kann man Folien für die Overheadprojektion zu Vorträgen oder – wie hier – zur Illustration von Artikeln erzeugen. Es handelt sich dabei zwar bisher nicht um ausgewachsene CAD-Programme, aber für illustrative Zwecke eigenen sie sich hervorragend.

Da gibt es die verschiedensten grafischen Objekte, die man einfach auf einer Menüleiste auswählt. Man kann die Grafikobjekte auf der imaginären Zeichenfläche des Bildschirms beliebig umherschieben, vergrößern, verkleinern und verzerren. Anschließend lassen sie sich mit verschiedenen Linienarten umranden und mit einer Vielzahl attraktiver Muster ausfüllen. Ja, man kann sie sogar übereinander legen und sortieren, das heißt bestimmen, welches Objekt im Vordergrund, welches im Hintergrund liegen soll. Schließlich kann man eine Gruppe von Objekten bilden, die sich dann wie ein Gegenstand verschieben, vervielfachen und vergrößern oder verkleinern lassen.

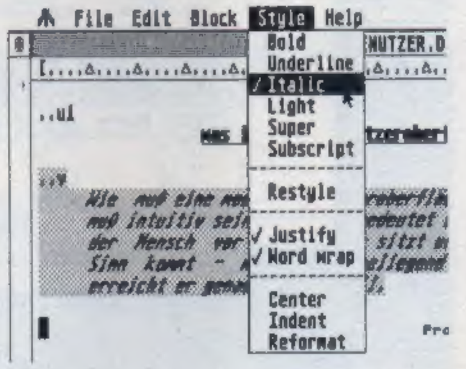
Dasselbe kann man tun, wenn man einige Dateien von einer Diskette auf eine andere kopieren will. Die Disketten werden bei einem Computer durch Karteikästen dargestellt, die auf ihnen gespeicherten Dateien als Stapel von Blättern (Dokumente) oder durch andere Symbole. Man fängt einfach die gewünschten Objekte mit einem Lasso ein. Anschließend können sie gemeinsam verschoben werden.

Alle diese Dinge will ich die magische Welt der Mediencomputer nennen. Ohne bestimmte technische und programmmäßige Voraussetzungen kann diese magische Welt nicht inszeniert werden. Fangen wir mit der Hardware an, also dem, wogegen man klopfen kann (daß es in der magischen Welt auch mal klingelt, wenn man mit dem Mauszeiger danebentrifft, ändert nichts daran, daß das dann Software ist).

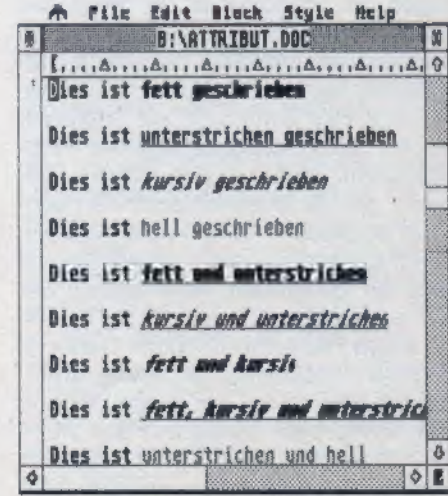
Physikalisch gesehen wird die Benutzeroberfläche durch Bildschirm, Tastatur und Maus dargestellt. Über die technischen Daten erfahren Sie an anderer Stelle in diesem Heft Genaueres. Hier geht es um die ergonomischen Gesichtspunkte, die unmittelbar mit der Bedienung der Computer und der Wahrnehmung der magischen Welt zu tun haben.



Anklicken: Man deutet mit dem ***Mauszeiger*** auf ein Objekt und drückt kurz die Maustaste. Nun ist das Objekt aktiviert (farblich hervorgehoben), es ist »das Objekt, das wir meinen«.



Anwählen: In einem ***Pull-Down-Menü*** wird eine Arbeismöglichkeit gewählt, indem man den Mauszeiger auf den entsprechenden Eintrag bringt. Dieser wird farblich hervorgehoben. Je nach System muß man nur die Maustaste drücken beziehungsweise sie loslassen. Braucht man das Menü nicht mehr, bewegt man den Zeiger aus dem Feld und drückt einmal den Knopf.



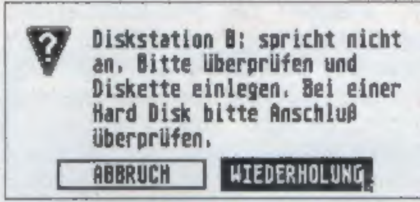
Aufzug (proportionaler): Der Aufzug zeigt an, welche Stelle eines Dokumentes (Text, Inhaltsverzeichnis) gerade angezeigt wird. Er kann auch benutzt werden, um andere Stellen zur Anzeige zu bringen, indem man ihn verschiebt.

Manche der Mediencomputer haben nur Schwarzweiß-Bildschirme (Macintosh), andere farbige (Amiga), bei einem kann man (bisher) wählen (Atari).

Ergonomie: Qualität von Hard- und Softwarekonzepten

Für die Qualität der Benutzeroberfläche ist es nicht ausschlaggebend, daß man möglichst viele bunte Bilder sieht und spektakuläre Töne hört. Es kommt auf die arbeitsgerechte Gesamtfunktion, die Ergonomie, an. An erster Stelle steht der Bildschirm. Er muß absolut scharf und flimmerfrei sein. Dabei werden an die Bildschirme der Mediencomputer höhere Anforderungen als an die Bildschirme von Bürocomputern gestellt. Da die Mediencomputer mit sehr hohen Auflösungen (640 Punkte oder mehr horizontal, 400 Punkte oder mehr vertikal) arbeiten, um feine grafische Symbole, Sonderzeichen und verkleinerte Schriftzeichen darzustellen, muß die Röhre über die gesamte Bildfläche absolut scharf abbilden. Achten Sie darauf, daß auch hoch- oder tiefgestellte Schrift noch einwandfrei zu lesen ist – auch am Rand des Bildschirms. So sehr Farbe eine Bereicherung darstellt – Farbbildröhren fügen sich entweder nicht in das jetzige Preisschema, oder sie haben eine verringerte Auflösung (um 320 mal 200 Punkte). Wenn, was wünschenswert ist, die Schrift schwarz auf weiß, wie auf einem richtigen Blatt Papier, dargestellt wird, muß der Bildschirm eine höhere Bildwechselfrequenz als üblich haben. Der Grund ist, daß die Flimmergrenzfrequenz des Auges um so höher ist, je heller das Licht. Ein weißer Bildschirm mit dunklen Zeichen hat aber eine höhere Grundhelligkeit als ein schwarzer mit hellen Zeichen. Hier enttäuscht der Amiga mit seinem sonst ausgezeichneten Farbbildschirm. Man hat sich mit der Frequenz an die amerikanische Fernsehnorm angepaßt – wohl in der Absicht, Computergrafik und Video-Signale mischen zu können. Aber setzen Sie sich einmal eine Stunde 40 bis 50 cm vor einen Fernsehbildschirm! Sehr gut sind die Bildschirme von Apple, ebenfalls ausgezeichnet der große, scharfe und flimmerfreie Bildschirm des Atari. Der Mac-Bildschirm wirkt deutlich schärfer, das liegt wohl auch daran, daß ein kleines Fernsehbild, aus der Nähe betrachtet, schärfer aussieht als ein großes. Da es beim Atari kein Computergehäuse gibt, auf das man den Bildschirm stellen kann, finde ich ihn zu tief stehend. Es fehlt eine Verstellmöglichkeit. Man kann ihn auf einen Bücherstapel stellen oder durch Unterlegen einer Diskettenschachtel unter die Vorderkante aufrichten. Ähnlich unpraktisch sind die Laufwerke, mindestens das untere liegt mit seinem Einwurfschlitz so nahe an der Tischoberfläche, daß dies unhandlich ist.

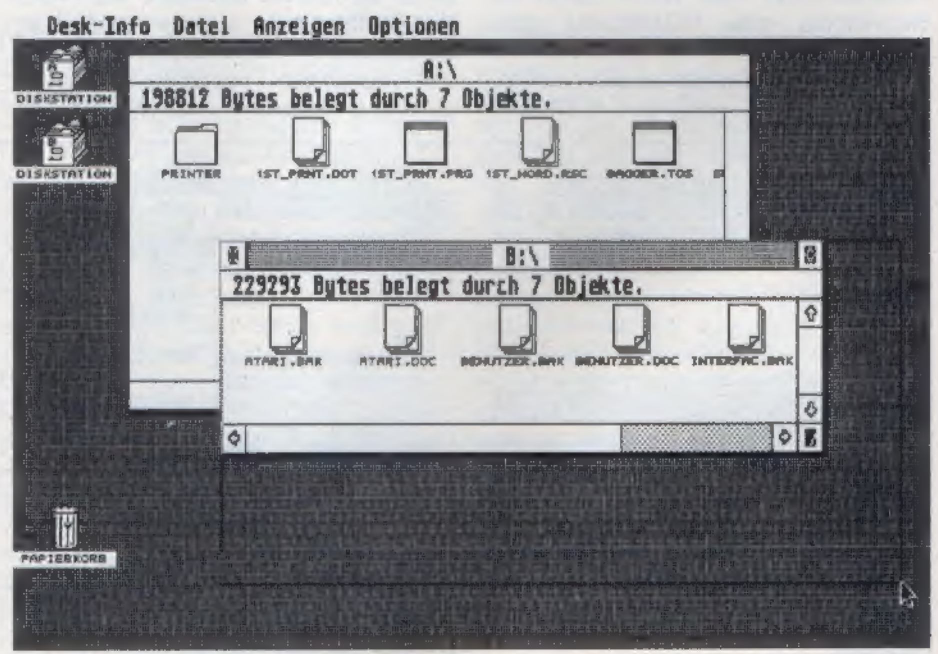
Der ST ist eindeutig besser zum Schreiben geeignet als ein IBM-PC. Während ich sogar bei meinem Ganzseitenbildschirm zum Korrekturlesen von Manuskripten einen papierenen Ausdruck vorziehe, weil das Lesen negativer, primitiv und eng dargestellter



Dialogbox: Ein Kasten, in dem eine Nachricht an den Benutzer steht. Eventuell kann er Antworten eingeben oder über ***Schalter*** eine Auswahl treffen. Dieser Bildschirmausschnitt dient also der Verständigung mit dem Computer.

Dokumente: Gespeicherte Informationen, die ein abrufbares Wissen enthalten (Text, Grafik). Werden meist in einer Form dargestellt, die auf ihren Charakter hinweist, zum Beispiel mehrere Blätter für Texte.

Doppelklick: Man deutet mit dem ***Mauszeiger*** auf ein Objekt und drückt in einer reflexartig-schnellen Geste sehr schnell zweimal die Maustaste. Dadurch sagen wir, daß wir mit diesem Objekt etwas tun wollen – und es soll gleich getan werden. Alle Arten von Aktivitäten können damit ausgelöst werden, ohne daß irgend eine weitere Aufforderung an den Computer notwendig ist. Disketten-Fenster werden geöffnet, Programme direkt gestartet, Files gelöscht, kopiert und vieles andere mehr.



Fenster: Ausschnitte auf dem Bildschirm, die bestimmten Dokumenten oder Programmen, die gerade laufen, zugeordnet sind. Meist weiß unterlegt mit doppelten Rändern, die verschiedene Schalter enthalten.

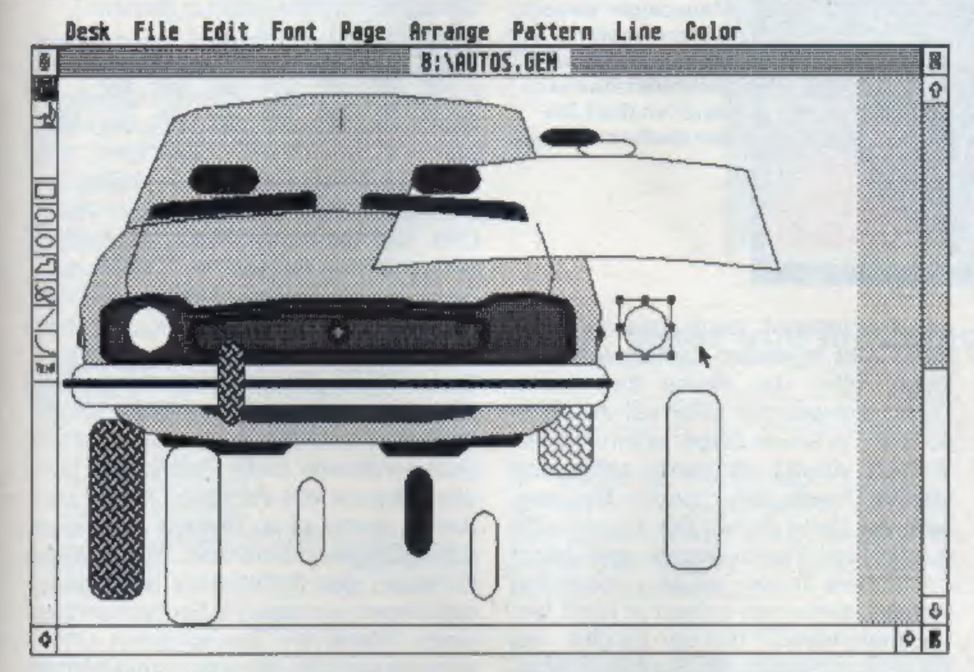
Karteikästen: Symbol für ein Arbeitsmittel, das verschiedene Informationen aufbewahrt (Diskette Massenspeicher). Durch ***Doppelklick*** wird ein Karteikasten geöffnet, in dem erscheinenden Fenster sieht man seinen Inhalt.

Schrift vom Bildschirm die Augen anstrengt, kann ich auf dem Atari sehr gut auch längere Texte lesen. Der ruhige Bildschirm, die große, klare Schrift (schwarz auf weiß) mit einem ausreichend Abstand zwischen den Zeilen haben sogar eine bessere Qualität als die von üblichen Matrixdruckern auf Papier. Während der Amiga-Monitor spiegelt, als wäre er eigens poliert, hat der Ataribildschirm eine hochwertige Antireflex-Ätzung. Man kann eine Schreibtischlampe leicht streifend über das Gehäuse des Bildschirms und die Tastatur leuchten lassen, ohne daß dies stört. Dadurch erhält man sehr natürliche Lichtverhältnisse.

Will man den Computer beruflich verwenden – ob nun im Büro, als Schriftsteller oder Grafiker – so kommt an zweiter Stelle, aber kaum weniger wichtig, die Qualität der Tastatur. Es ist ein Irrtum, zu glauben, wenn man eine Maus hat, bräuchte man keine Volltastatur mehr. Schlimmes Beispiel in dieser Richtung: Apples Macintosh. Die Tastatur ist klein, aber dennoch klobig, sie besitzt weder Funktionstasten noch Cursortasten, auch keinen Nummernblock.

Schon wesentlich besser nimmt sich die Atari ST-Tastatur aus. Es ist eine professionelle Normtastatur mit gutem Layout (abgesehen von den verspielten Funktionstasten, die eher an Rallyestreifen am Auto eines Spoilerfreaks erinnern). Sie hat zwei abgesetzte Blöcke, ganz rechts Rechentasten mit Klammern, den vier Grundrechenzeichen, Dezimalpunkt und eigener ENTER-Taste (die programmtechnisch von der RETURN-Taste unterschieden werden kann). Zwischen diesem Block und der Haupttastatur befinden sich kreuzähnlich angeordnete Cursortasten mit vier weiteren Editor-Funktionstasten. Eine Funktionsanzeige der Großbuchstaben-umschaltung fehlt leider. Von allen hier erwähnten Computern hat die Atari-Tastatur die reichhaltigste Ausstattung. So gäbe es an dieser Tastatur wenig auszusetzen, wäre die Qualität der Tasten nicht so schlecht. Sie sind klapprig und wackelig, der Anschlag ist unsicher – fällt er ein wenig zu leicht aus, so sprechen die Tasten nicht an. Auch die Abfrageroutine der Tastatur ist mangelhaft, sie hat kein N-Key-Rollover: Drückt man mehrere Tasten nacheinander, ohne die ersten wieder losgelassen zu haben, so sprechen Zeichen nicht an, in manchen Programmen werden auch völlig falsche Steuercodes ausgelöst. So wird eine schnell blindschreibende Person keine ungetrübte Freude haben. Da die Tastatur auch den eigentlichen Computer enthält, ist sie schwer, nach hinten sehr umfangreich und – wegen der zahlreichen dicken Kabel – praktisch unbeweglich. Trotzdem ist der Atari mit einem entsprechenden Programm recht gut als Textsystem zu gebrauchen – wenn man sich auf die Tastatur einstellt und gelegentliche Korrekturen in Kauf nimmt.

Passabel ist die Tastatur des Amiga – die Tasten haben einen gut fühlbaren Druckpunkt, die Funktionstasten am oberen Rand sind etwas verbreitert und in zwei Gruppen gegliedert. Darüber befindet sich ein eingesenkter Rahmen, in dem man austauschbare Scha-

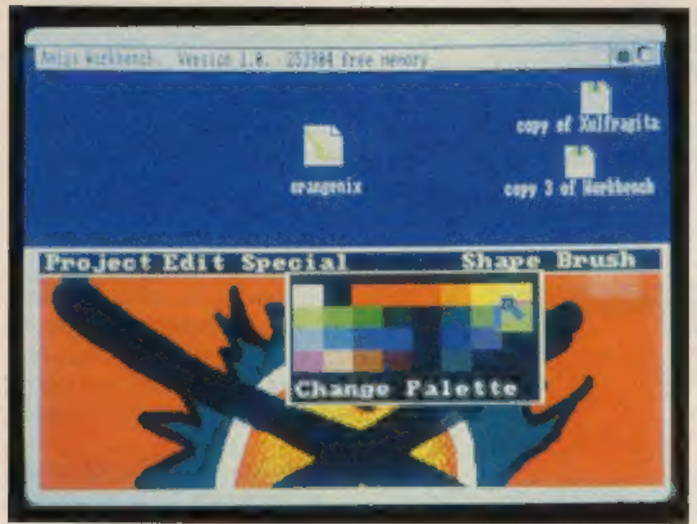


Grafik: Der Computer kann beliebige zeichnerische Darstellungen erzeugen, die auf der Zusammensetzung aus grafischen Grundobjekten beruhen. Computer gespeichert.

Objekte: Scheinbar reale Gegenstände in der ***magischen Welt*** des Computers, die man bewegen oder für verschiedene Zwecke benutzen kann. Diese können mit verschiedenen Mustern versehen und übereinandergelegt werden.

Gruppe von Objekten: Verschiedene in der ***magischen Welt*** existierende Gegenstände oder grafische Grundobjekte können zu einer Ganzheit (einem zusammengesetzten Gegenstand) vereint werden. Ein solches zusammengesetztes Objekt kann als Ganzes verschoben oder in der Größe verändert werden.

Desk File Edit Font Page Arrange Pattern Line Color
B:\AUTOS.GEM



Durch Druck auf die rechte Maustaste wechselt die Anzeige in der oberen Fensterkante von »Orangenix« in verschiedene Menümöglichkeiten. Auf eine wurde der Mauszeiger bewegt, dadurch sprang eine Jalousie mit mehreren Auswahlpunkten (hier Zeichenfarben) heraus.

bionen mit Funktionsbeschriftungen unterbringen kann. Die Cursortasten sind zwar nicht von der Haupttastatur abgesetzt, aber vorbildlich in Form eines Kreuzes angeordnet. Der Rechenblock rechts ist nicht so gut ausgestattet wie beim Atari, außer dem Minus fehlen die Rechenzeichen, nur ein Dezimalpunkt und eine ENTER-Taste sind vorhanden. Die Großbuchstaben-Feststelltaste hat eine LED-Anzeige; ähnlich wie bei Apple findet man links und rechts neben der Leertaste je eine »Amiga«-Taste. Für Blindschreiber sind die Positionen der Zeigefinger mit feinen Fühlmarken auf den Tasten »F« und »J« versehen.

Apple meint also, man brauche keine Sondertasten, weil man ja die Maus hat (trotzdem gibt es links und rechts von der Leertaste je zwei Spezialtasten mit zunächst unerfindlicher Bedeutung). Manche Benutzer fühlen sich dadurch bevormundet. Der Atari macht hier keine Kompromisse – man kann in Textsystemen bestimmte Arbeitsweisen auf Funktionstasten legen. Eine gute Lösung ist die Anzeige dieser Tasten am unteren Bildschirmrand wie bei 1St-Word, dabei können die Funktionen dieser imaginären Tasten sowohl mit der Maus »gedrückt«, physisch über die Tastatur eingegeben oder über Pulldown-Menüs ausgelöst werden – in jedem Fall zeigen sie den jeweiligen Funktionszustand an (schwarz für aktiv oder weiß für inaktiv). Jeder kann diesen Computer also entsprechend seinen persönlichen Vorlieben benutzen.

Haben Sie schon einmal beigewohnt, wenn jemand etwas erklärt, das sich auf dem Bildschirm eines Computers abspielt? Leute drängen um die ach so kleine Flimmerscheibe. Entweder weiß man nicht, was der Vortragende mit seinen Worten meint, oder ein klobig deutender Finger verdeckt das Wesentliche.

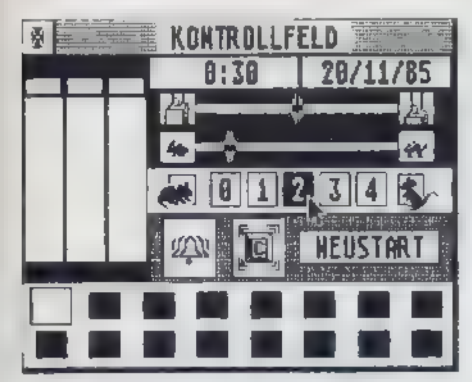
Wieviel eleganter dagegen mit der Maus. Man sitzt nicht vorgebeugt, sondern entspannt zurückgelehnt. Die Hand ruht irgendwo seitlich auf dem Tisch, führt das kleine Instrument. Freier, entspannter Blick auf den Bildschirm. Ein flinker Zeiger untermalt den Vortrag, deutet eindeutig selbst auf feinste Details, auf einzelne Buchstaben, auf den i-Punkt. Die Maus sollte gut in der Hand liegen und leicht bedienbare Tasten haben – möglichst wenige, denn man schaut ja nicht hin, und mehr verwirrt hier nur. Es gibt – bis heute – nur eine gute Sorte von Mäusen: mechanische. Lassen Sie sich nichts einreden – sogenannte optische Mäuse sind nicht das Gelbe vom Ei. Es würde hier zu weit führen, den Grund zu erklären. Immerhin soviel: Eine Hand ist ja auch etwas Mechanisches, und eine mechanische Maus hat kein starres Koordinatensystem, sondern fährt, wohin die Hand will. In dieser Hinsicht sind die Mäuse der drei Systeme gleich gut – es scheint sogar, daß die innere Mechanik aller drei Mäuse, trotz recht unterschiedlichem äußeren – vom selben Hersteller stammen.

Ein weiterer interessanter Gesichtspunkt ist der Übersetzungsfaktor der Maus (eine Frage der Software). Je kleiner er ist, das heißt, je weiter man die Maus auf dem Tisch verschieben muß, um eine bestimmte Bewegung des Mauszeigers auf dem Bildschirm zu erreichen, desto mehr Platz braucht man auf dem Tisch. Dafür aber läßt sich der Mauszeiger präziser positionieren. Apple verwendet einen relativ großen Faktor, der Zeiger bewegt sich auf dem Bildschirm rascher und weiter als die Maus auf dem Tisch. Dadurch kann man den Zeiger mit weniger Bewegungsaufwand rasch über den ganzen Bildschirm positionieren, aber man hat manchmal Schwierigkeiten, feine Details wie Buchstaben zu treffen. Angenehmer ist eine annähernde 1:1-Übersetzung wie beim Atari.

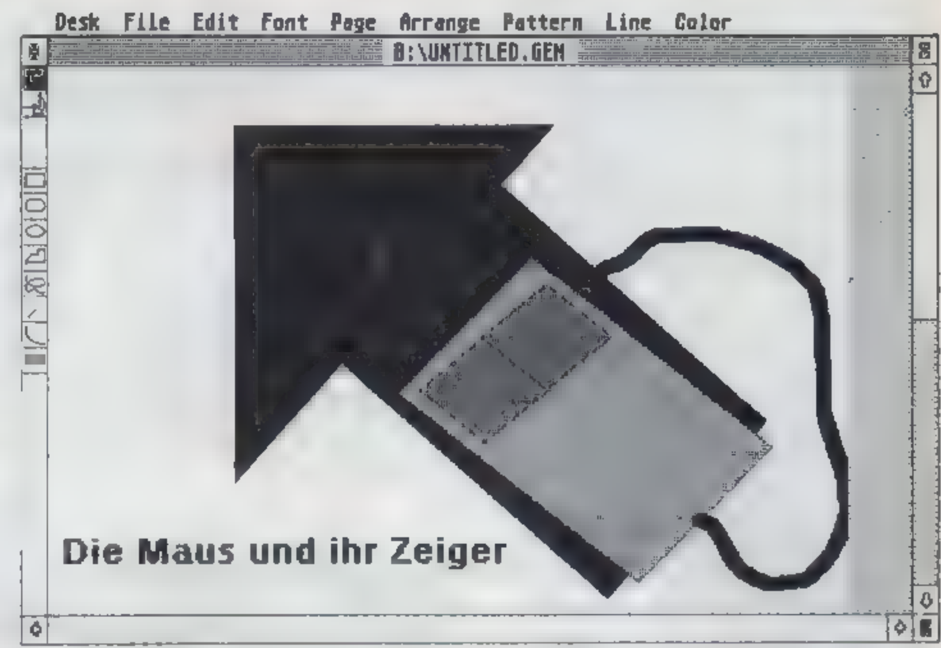
Wie Apple ja beim Mac betont, gehören auch die physischen Abmessungen des Computers zu den ergonomischen Merkmalen. Hier schneidet der Atari am schlechtesten ab – man konnte sich nicht von den bei Heimcomputern üblichen, praktisch unbegrenzt vielen Kästchen und Netzteilen, alles mit viel Kabelwerk verbunden, trennen. Es gibt preiswerte Bausätze, um alles in einem Gehäuse verschwinden zu lassen.

Sehr umfangreich ist – trotz seiner zierlicheren Tastatur – der Amiga. Der Computer besteht aus einem Grundgehäuse und einem großen Bildschirm, den man obendrauf stellen kann. Ein Laufwerk ist eingebaut, ein zweites findet neben dem Hauptgehäuse Platz. Und man braucht nur zwei Netzkabel (eines für den Bildschirm). Damit erfordert dieser Computer annähernd so viel Platz wie ein IBM-PC. Ich möchte aber betonen, daß ich eine erwachsene Tastatur und eine professionell gestaltete Maschine mit Erweiterungsmöglichkeiten vorziehe, auch wenn das ein paar Zentimeter mehr Platz kostet. Auf eine Vielzahl von Personen macht der Amiga, wenn er in Betrieb ist, einen unangenehmen Eindruck. Das liegt am Flimmern des Bildschirms und daran, daß dieser, wie manche Fernsehgeräte, einen intensiven Ton im tiefen Ultraschall ausstrahlt, der nicht direkt hörbar ist, aber Kopfdruck erzeugt. Ich kann mit dem Computer nicht länger als ein paar Minuten arbeiten. Allerdings ist die Empfindlichkeit dagegen individuell verschieden.

Ein sehr wesentlicher Punkt ist die Arbeitsgeschwindigkeit. Es gibt viele Untersuchungen darüber, wie Menschen auf unterschiedliche Antwortzeiten von Softwaresystemen reagieren. Wer seinen Heimcomputer als Hobby betreibt, mag es hinnehmen, wenn dieser sekundenlang oder auch Minuten braucht, um irgendeinen Befehl zu verarbeiten oder ein Programm aufzurufen. Auf den Mediencomputern verlangt jede Aktion in der magischen Welt ein hohes Maß an Prozessorleistung. Wir erwarten aber, daß Werkzeuge, mit denen wir umgehen, sich in ihrem Zeitverhalten der menschlichen Reaktions- und Entscheidungszeit anpassen – das sind nur Sekundenbruchteile. Stellen Sie sich einmal vor, Sie müßten beim Lesen eines Buches am Ende einer Seite immer drei Sekunden warten, weil sich die Blätter nicht schneller umwenden lassen. Daher brauchen diese Computer auch so leistungsfähige Prozessoren und eventuell noch Coprozessoren. Je schneller ein Softwaresystem antwortet, desto eher kann es in seiner Handhabung mit dem Menschen verwachsen und zu einem verlängerten Sinnesorgan werden. Je langsamer es reagiert, desto eher erscheint es uns als ein gegen uns agierender Gegen-

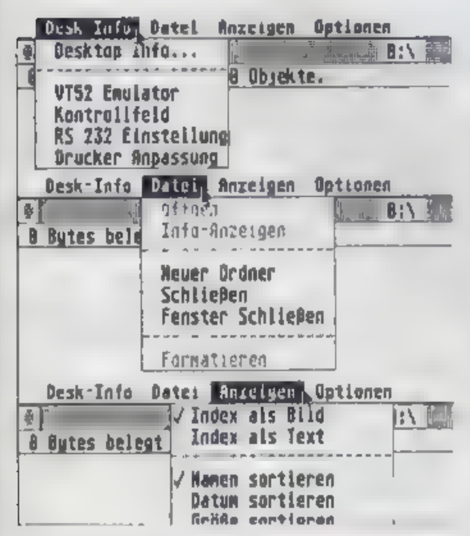


Knöpfe: Schalter in der magischen Welt, mit denen Befehle gegeben oder bestimmte Zustände eingestellt werden können. Hier zum Beispiel die Reaktionszeiten von Tastatur und Maus.

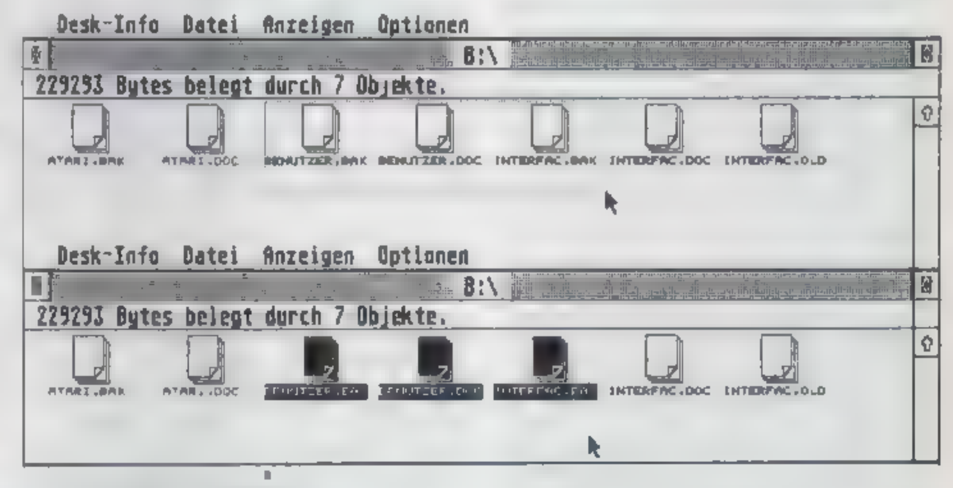


▲ ***Maus:*** Ein mechanisches Hilfsmittel, das mit der Hand geführt wird. Auf dem Bildschirm folgt der Bewegung ein ***Zeiger*** oder ***Symbol***, das verschiedene Formen annehmen kann.

Lasso: Ein magisches Band, mit dem verschiedene Gegenstände beziehungsweise Symbole der magischen Welt zu einer ***Gruppe*** zusammengefaßt werden können. ▼



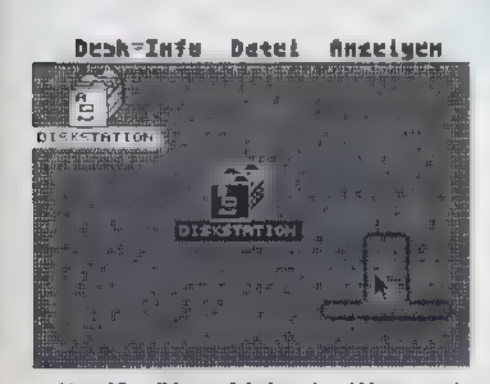
Menüleiste: Eine grafisch abgesetzte Leiste, auf der verschiedene Arbeitsmöglichkeiten oder Befehle an den Computer aufgezählt sind, aus denen man durch Deuten mit der Maus auswählen kann. Dabei springt ein ***Pull-Down-Menü*** mit verschiedenen Unterwahlmöglichkeiten wie eine Jalousie herunter.



▲ ***Schalter → Knöpfe***
Starten eines Programms → Doppelklick

Symbole: → Objekte in der magischen Welt werden durch Bildsymbole dargestellt. Hier Dateien und Programme.

Text und Grafik: Einige Programme erlauben es bereits, mit Zeichenprogrammen erstellte Grafiken in eine Textverarbeitung einzubeziehen. ▼



Umriß (Verschieben): Wenn ein Objekt verschoben wird, so folgt dem Mauszeiger ein Schatten, der die Position des Objektes nach Abschluß des Vorgangs andeutet.

hello

Herr Leckebusch,

ich schreibe immer noch auf dem , weil auf dem ST solche Gags (noch) nicht möglich sind.
Ansonsten, habe heute gerade Ihre Disks erhalten und Superfilter kurz angespielt. Positiv war festzustellen, daß es auch bei großen Files nicht aussteigt, was ich nicht von allen ST-Programmen behaupten kann. Werde weiter betatesten und melde mich beim ersten Bug.

stand. Besonders störend ist es, wenn Textsysteme zu langsam reagieren.

Die wichtigsten und häufigsten Handgriffe (etwa das Aktivieren eines Fensters) sollten also höchstens wenige Sekundenbruchteile dauern, das Starten von Programmen höchstens Sekunden. Das seitenweise Blättern in Texten sollte blitzartig geschehen, das Suchen oder Blättern über größere Entfernungen möglichst auch nicht mehr als ein paar Sekunden beanspruchen. Beim Schreiben sollten die Buchstaben erscheinen, ohne nachzuhinken, auch wenn der Editor gerade einen Wortumbruch auf eine neue Zeile vornimmt. Beim Rollen des Textes soll dieser starr wie auf einer Folie vorbeiziehen und nicht sichtbar neu geschrieben werden. Am besten ist es, wenn die Reaktionszeiten so kurz sind, daß man sie nicht mehr wahrnimmt...

File Edit Block Style Help

B:\APFEL.DOC

Äpfelein, Äpfelein auf dem Schirm...

Seit in der Oktobernummer 1985 von *Spektrum der Wissenschaft* eine genaue Beschreibung des Algorithmus erschien, mit dem man auf einem Computer die sogenannte >>Apfelmännchen<<-Grafik erzeugen kann, wurde die Welt von einer Fülle von Apfelmännchen-Programmen überschwemmt.

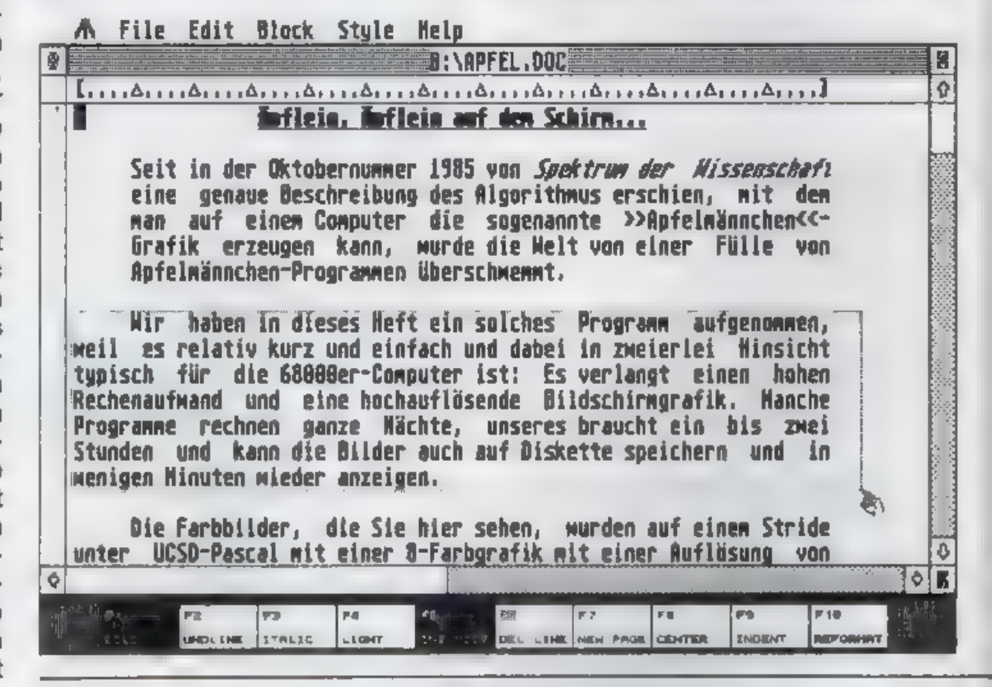
Wir haben in dieses Heft ein solches Programm aufgenommen, weil es relativ kurz und einfach und dabei in zweierlei Hinsicht typisch für die 68000er-Computer ist: Es verlangt einen hohen Rechenaufwand und eine hochauflösende Bildschirmgrafik. Manche Programme rechnen ganze Nächte, unseres braucht ein bis zwei Stunden und kann die Bilder auch auf Diskette speichern und in wenigen Minuten wieder anzeigen.

Die Farbbilder, die Sie hier sehen, wurden auf einem Stride

F2 UNDLINE | F3 ITALIC | F4 LIGHT | F6 DEL LINE | F7 NEW PAGE | F8 CENTER | F9 INDENT | F10 REFORMAT

▲ ***Tasten, imaginäre:*** Symbole auf dem Bildschirm, die in der magischen Welt die Funktion von Tasten haben. Siehe ***Knöpfe***. Hier 1ST Word auf dem Atari.

Textabsatz: Ein Textabschnitt kann durch Überfahren mit dem ***Mauszeiger*** angewählt und anschließend in einer anderen Schrift dargestellt oder verschoben werden. ▼


File Edit Block Style Help

B:\APFEL.DOC

Äpfelein, Äpfelein auf dem Schirm...

Seit in der Oktobernummer 1985 von *Spektrum der Wissenschaft* eine genaue Beschreibung des Algorithmus erschien, mit dem man auf einem Computer die sogenannte >>Apfelmännchen<<-Grafik erzeugen kann, wurde die Welt von einer Fülle von Apfelmännchen-Programmen überschwemmt.

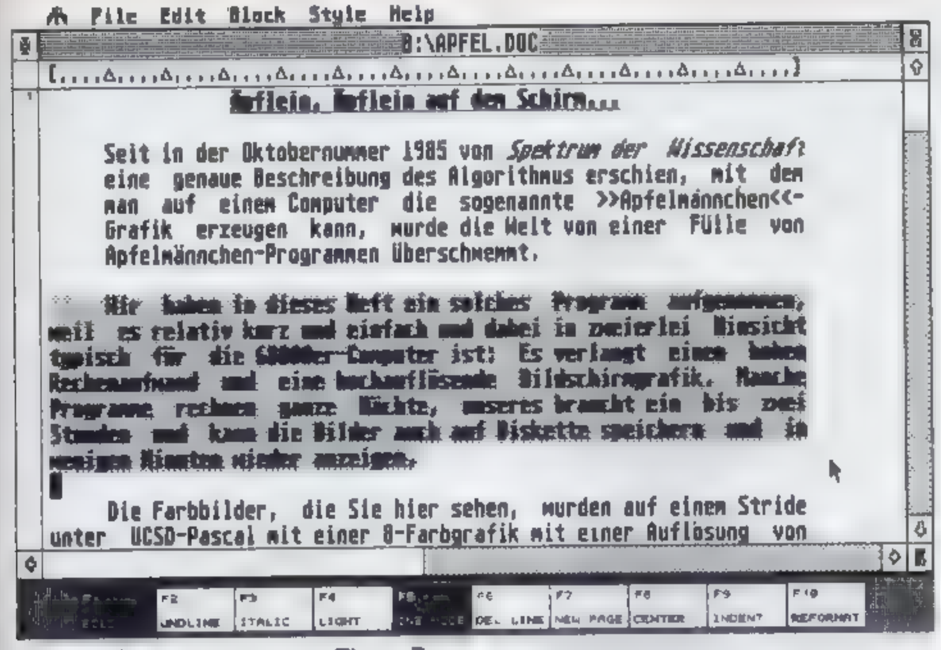
Wir haben in dieses Heft ein solches Programm aufgenommen, weil es relativ kurz und einfach und dabei in zweierlei Hinsicht typisch für die 68000er-Computer ist: Es verlangt einen hohen Rechenaufwand und eine hochauflösende Bildschirmgrafik. Manche Programme rechnen ganze Nächte, unseres braucht ein bis zwei Stunden und kann die Bilder auch auf Diskette speichern und in wenigen Minuten wieder anzeigen.

Die Farbbilder, die Sie hier sehen, wurden auf einem Stride unter UCSD-Pascal mit einer 8-Farbgrafik mit einer Auflösung von

F2 UNDLINE | F3 ITALIC | F4 LIGHT | F6 DEL LINE | F7 NEW PAGE | F8 CENTER | F9 INDENT | F10 REFORMAT

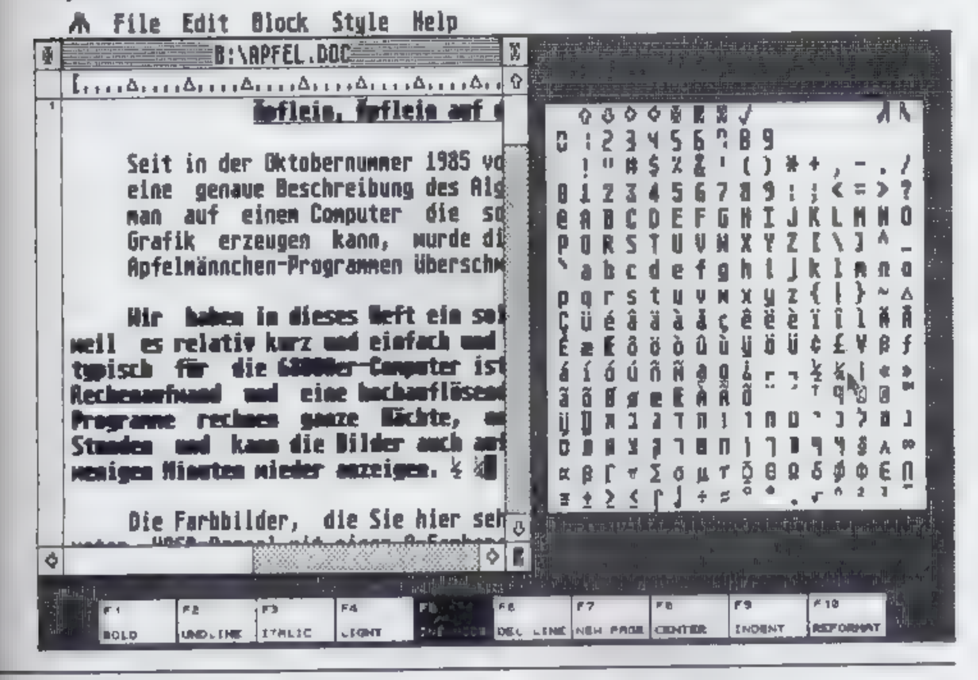
Mac – Lisas Erbe

Apple hat zwar die Fenster, die Maussteuerung und die Objektkonzepte nicht erfunden – das war Xerox. Doch hat Apple für das Design seiner Benutzeroberfläche intensive Forschungen betrieben, während die Konkurrenten, so hat man manchmal das Gefühl, nur bei Apple abgeschaut haben. Bei Apple steht die Bedienungs-Sicherheit an allererster Stelle. Die Software soll die Gewalt über die Maschine haben und den Benutzern vor Fehlern jeder Art bewahren. Wirft man auf dem Mac etwas in den Papierkorb, so kann man es (nur den letzten Wegwurf) wie aus einem richtigen Papierkorb wieder herausholen, falls es sich um ein Versehen handelte. Man kann dem Mac auch nicht einfach eine Diskette wegnehmen, sondern ihn nur auffordern, sie herauszugeben. Der Computer sichert dann eventuell auf dem Schreibtisch herumliegende, sprich nur im Arbeitsspeicher befindliche, Arbeitsergebnisse automatisch, ehe er die Diskette auswirft. (Beim Mac hat man außerdem ein Löchlein vorgesehen, in das man mit einer aufgebogenen Büroklammern, hineinstoßen kann, um ihn zur Herausgabe einer Diskette zu zwingen). Der Atari macht diesen feinen Unterschied gar nicht – der Benutzer ist selbst dafür verantwortlich, mit einem Editor bearbeitete Texte vor dem Ausschalten abzusichern und dafür die richtige Diskette einzusetzen. Man kann beim Atari nur die Anordnung von Fenstern auf dem Bildschirm abspeichern, während Lisa und Mac nach dem Einschalten den Schreibtisch wieder genau so vollräumen, wie man ihn beim Ausschalten hinterlassen hat. Man kann auf dem Mac Dokumente, also Programme oder Dateien, aus den Disk-Fenstern beziehungsweise den Ordnern herausnehmen und direkt auf den Schreibtisch legen, um dann das die Sicht verdeckende Fenster wieder zuzumachen. Auf dem Mac bleiben Dokumente in den Fenstern auch an der Stelle liegen, an der man sie abgelegt hat – so daß sie nicht aus den Augen verschwinden. Nur auf Anforderung sortiert der Mac Dokumente und ordnet sie in Reih und Glied ein. Während man beim Mac Dialogboxen, ehe man sich daran macht, die von ihnen gestellten Fragen zu beantworten, beiseiteschieben kann, sind sie beim Atari starr. Wenn sie – was in manchen unausgereiften Programmen schon mal vorkommt – eine wichtige Information verdecken, hilft nichts. Während MacWrite verschiedene Fonts (Schriftarten und -Größen) anbietet und Text und Grafik mischen kann, gab es zum Redaktionsschluß auf dem Atari nur ein Programm, das mit einer Schriftart und den Hervorhebungen fett, unterstrichen, kursiv, grau und Hoch- beziehungsweise Tiefstellung arbeitet. Aber es ist sicher nur eine Frage der Zeit, bis der ST hier aufholen wird. Für den Amiga gab es zum



▲ ***Zeichenprogramm:*** Ein Programm, mit dem man malen oder geschäftlich/technisch/wissenschaftliche Darstellungen erzeugen kann. Man kann unter verschiedenen grafischen Grund-***Objekten*** wählen.

Zeichentafel: Magische Zusatztastatur in einer Textverarbeitung, mit der ausländische oder technisch/wissenschaftliche Zeichen eingegeben werden können. Die Auswahl ist sehr groß. ▼



gleichen Zeitpunkt noch gar keine richtige Textverarbeitung.

Atari hat darauf verzichtet, eine eigene Benutzerumgebung zu entwickeln – stattdessen hat man das geräteunabhängige Softwarekonzept von Digital Research (DR) übernommen. Ich glaube, daß das – trotz mancher Einschränkungen – eine gute Wahl war. Im letzten halben Jahr ist die Software recht stabil geworden. Lisas Desktop-Manager (Schreibtisch-Verwalter) war bei der ersten Präsentation des Computers sehr fehlerhaft und langsam. GEM ist dagegen sehr schnell. Außer den Standardprogrammen gab es für die Lisa lange Zeit kaum Software. Beim Schreiben dieser Artikel und verschiedener Programme habe ich mehrere Monate mit einem Atari gearbeitet und den Eindruck gewonnen, daß er sehr zuverlässig ist.

Wichtig ist beim Atari, daß GEM auf einem Betriebssystem aufsetzt, auf dem man auch konventionell programmieren kann. Das ist dann nicht schwieriger als etwa unter MS-DOS (der Bagger und Superfilter von Computer persönlich konnten so innerhalb weniger Wochen entwickelt werden).

DR hat seine magische Welt ziemlich genau bei Apple abgeschaut (darüber kam es dann auch zu juristischen Auseinandersetzungen). Aus der Sicht des Benutzers halte ich diese Anlehnung für einen Vorteil – es ist sehr leicht, von der Bedienung eines Mac auf den Atari umzusteigen – dagegen muß man sich auf dem Amiga umstellen. Darüber hinaus hat man einige Veränderungen, aber auch Vereinfachungen der Konzepte vorgenommen.

Zu den Gemeinsamkeiten der Welten von Apple und DR gehört die Manipulation von Fenstern und Pictogrammen. Man kann Fenster verschieben, in der Größe verändern, sie auf- und zumachen. Verschiebt man ein Objekt, so folgt dem Zeiger der Maus ein geisterhafter Umriß und nicht das voll gezeichnete Fenster. Dies geschieht aus Geschwindigkeitsgründen – das Neuzeichnen eines Fensters mit seinem detaillierten Inhalt, wobei andere Objekte verdeckt oder aufgedeckt werden können, kann nicht in Echtzeit erfolgen, es wäre zu schwerfällig. Man hat aber durch den sich bewegenden Umriß das Gefühl, daß man das Objekt oder jedenfalls seinen »Geist« tatsächlich in der Hand hat und ihn bewegt...

GEM – einfach und doch vielseitig

Bei den Pull-Down-Menüs findet man einen eher subtilen Unterschied: Bei Apple muß man einen Eintrag in der Menüleiste anklicken und die Maustaste festhalten, um die Jalousie mit den Wahlmöglichkeiten herunterspringen zu lassen. Die Wahlmöglichkeit, auf die man den Mauszeiger bewegt, erscheint schwarz hervorgehoben. Läßt man die Maustaste los, so wird die gerade aktivierte Anweisung ausgelöst. Dabei passiert es aber oft, daß man während des Bewegens der Maus mit dem Finger abrutscht und so ungewollt ein Programm aufruft.

DR wollte schlauer sein, hat den Dreh aber auch nicht ganz heraus: Sobald der Mauszeiger auf einen Eintrag in der Menüleiste gerät, springt eine Jalousie herunter. Um einen Eintrag zu aktivieren, muß man auf ihn fahren und die Maustaste drücken. Das ist eine gute Idee – dumm ist nur, daß man sehr oft versehentlich, wenn man etwas am oberen Bildschirmrand zu tun hat, auf die Menüleiste gerät. Schon springt ein wunderschönes Menü herunter – und um es loszuwerden, muß man irgendo außerhalb der Jalousie hinklicken. Dabei wäre die Lösung ganz einfach – die Jalousie sollte wieder verschwinden, wenn der Mauszeiger das umrahmende Kästchen verläßt.

Auf dem Atari aktiviert man, wie unter einem herkömlichen Computerbetriebssystem, ein Programm (Texteditor, Zeichenprogramm). Dieses Programm fragt dann nach der Datei, die man bearbeiten möchte. Auf dem Macintosh oder dem Amiga dagegen klickt man eine Zeichnung (das entsprechende Symbol) oder einen Text an. Daß dazu ein Zeichenprogramm oder ein Texteditor gestartet werden muß, braucht der Benutzer nicht zu wissen – es geschieht zwar, aber automatisch, sozusagen im Hintergrund. Apple und Commodore orientieren sich viel stärker an dem, was der Benutzer tun, womit er arbeiten will (nämlich mit einem Text, einer Zeichnung, einer Abrechnung), nicht an den Hilfsmitteln, die er dazu braucht. In dieser Hinsicht entspricht die magische Welt des Atari noch ziemlich deutlich einem herkömlichen Betriebssystem. Innerhalb von geschickt geschriebenen Programmen steht die magische Welt des Atari der von Apple aber in nichts nach.

Amigas Intuition

Der Amiga verwendet ein eigenes Betriebssystem (Intuition), dessen äußere Erscheinung stärker von der Apple-GEM-Linie abweicht. Die Maus hat zwei Tasten, und diese werden auch beide benutzt (der Atari hat zwar eine Maus mit zwei Tasten, aber alle mir bisher bekannten Programme verwenden nur die linke Taste, der Mac hat nur eine Taste auf der Maus). Beim Atari muß sich der Benutzer nur drei verschiedene Bedienungsweisen ein und derselben Taste einprägen: Einfaches Anklicken (einmal kurz drücken), Klicken, festhalten und verschieben, doppelt Klicken. Beim Mac kommt das Anklicken, Festhalten und Herunterziehen von Pull-Down-Menüs mit Auswahl durch Loslassen dazu. Auf dem Amiga gibt es das Anklicken, Festhalten und Verschieben, Doppelklicken sowie Menüumschaltung mit der rechten Taste und Auswahl aus dem Pull-Down-Menü durch Loslassen. Die linke Taste dient dazu, durch Doppelklick Fenster aufzumachen oder durch einfaches Anklicken und Festhalten umherzuschieben. Wie auf dem Macintosh löst das Umherbewegen des Zeigers, ohne daß man eine Maustaste drückt, noch keinerlei Aktionen aus, während auf dem Atari die Menüs von selbst herunterspringen, wenn der Mauszeiger sie berührt. Amiga-Programme zeigen in dem Rahmen am oberen Rand eines Fensters normalerweise nur allgemeine Informationen an – den Namen des Programms, rechts Symbole zum Wegschalten oder Zurückrufen des Fensters. Eine Menüleiste erscheint erst, wenn man die rechte Maustaste drückt. Dadurch wechselt die Anzeige am oberen Rand des aktiven Fensters, es erscheint eine Aufzählung möglicher Menüs.

Außerdem kann man Fenster mit einem besonderen Schalter in der Fensterecke in den Vordergrund oder Hintergrund schalten – auch eine eigene Idee. Auf Apple oder Atari kommt man an ein Fenster, das vollständig verdeckt ist, nicht heran – man muß erst das darüberliegende Fenster schließen oder wenigstens beiseiteschieben.

Auf dem Amiga ist nicht die Spitze des Zeigers, sondern das rückwärtige Ende des Zeigerstils aktiv, das heißt, man muß nicht mit der Spitze des Zeigers, sondern mit seinem Stumpf auf etwas deuten. Auch in einem anderen Punkt ist die Amiga-Umgebung nicht intuitiv: Will man ein Objekt verschieben, so verwandelt sich der Mauspfeil in einen merkwürdigen Kringel, der wie ein Jahrmarktsgebäck aussieht. Das Objekt selbst bleibt an Ort und Stelle – läßt man aber die linke Maustaste los, so springt es an den neuen Ort.

Abgesehen von solchen Äußerlichkeiten ist das Amiga-System aber eher auf oder über dem Niveau von Apple als dem von Atari. Man kann Objekte von einem Diskettenfenster herunternehmen und sie auf dem Schreibtisch liegenlassen (wie bei Apple, beim Atari ist das nicht möglich). Man kann daher ein Fenster aufmachen, etwas herausnehmen, es auf die Bildschirmoberfläche (Schreibtisch) legen und das Fenster wieder verschwinden lassen. Dadurch erscheint der Schreibtisch aufgeräumter. Wenn man eine Zeichnung bearbeiten will, klickt man wie auf dem Mac die Zeichnung an – nicht ein Zeichenprogramm. Objekte innerhalb von Fenstern bleiben da liegen, wo man sie hingelegt hat, sie werden nicht ohne den Wunsch des Benutzers sortiert. Das ist angenehm, weil sie dadurch realer wirken und nicht ungewollt aus den Augen schwinden.

Die großen Stärken des Amiga liegen in seiner Fähigkeit, mehrere Programme gleichzeitig aktiv arbeiten zu lassen, sowie in seinen vielseitigen Farb- und Toneffekten. Meinem Eindruck nach paßt dieser Computer mehr für's breite Publikum sowie kreative Leute, die mit spektakulären Effekten arbeiten wollen. Die kreativen Möglichkeiten, etwa für farbgrafisches Design, sind bisher unerreicht. Für trockene Arbeit wie das Schreiben von Texten ist – trotz anderer Einschränkungen – der Atari sicher besser geeignet. Gäbe es aber einen Amiga mit Schwarzweiß-Bildschirm, so würde das Urteil vielleicht anders ausfallen. (le)

Begriffe

Amiga:* *Mediencomputer von Commodore. Ein sehr aufwendiges System mit Farbgrafik, Klanggeneratoren und so weiter mit eigener Software »Intuition«.

Atari ST:* *Mediencomputer von Atari Corporation. Ein preiswertes System, dessen Benutzeroberfläche auf GEM (***Grafics Environment Manager***) von Digital Research (DR) beruht.

Bagger: Ein Programm, mit dem Dateien zwischen so verschiedenen Computern wie Atari, IBM-PC oder Z80-Systemen mit CP/M sowie dem UCSD-Betriebssystem übertragen werden können (auf Leserservice-Diskette). Siehe auch → ***Superfilter***.

Benutzeroberfläche → ***magische Welt***

Bildschirm: Ein Ausgabemedium. Auf dem Bildschirm (Braunsche Röhre ähnlich einer Fernsehröhre, schwarzweiß oder in Farbe) kann der Computer Schriftzeichen, aber auch Grafik anzeigen. Der Bildschirm spiegelt bei einem ***Mediencomputer*** unmittelbar den Inhalt eines Teiles des Arbeitsspeichers wider. Der Computer schreibt in diesen Speicherbereich Informationen in einer solchen Form, daß sie für den Menschen sinnvoll wahrnehmbar ist und das Aussehen von Schrift- oder Grafikzeichen beziehungsweise symbolischen Gegenständen (→ ***Symbole***) hat.

Directory: Computer legen, um Informationen wiederzufinden, auf Massenspeichern wie Disketten (Floppy-Disk) Inhaltsverzeichnisse an, die man sich als Text oder – auf den ***Mediencomputern*** – in Symbolform anschauen kann. Jeder Eintrag steht für eine Datei, eine zusammengehörige Menge von Informationen – zum Beispiel für einen Brief (Textdatei).

ENTER → ***RETURN***

Ergonomie: Lehre von der menschengerechten Gestaltung von Arbeitsplätzen und Arbeitsmitteln (Bürostühle, Tische, Computertastaturen, Bildschirme, Software).

File: Computer speichern Informationen in technisch zusammenhängenden Datenstrukturen, die Files genannt werden. Zum Beispiel

können in der Textverarbeitung einzelne Briefe als je ein File gespeichert werden. Jedes File hat einen Eintrag im Inhaltsverzeichnis (→ ***Directory***)

GEM: Softwaresystem von Digital Research, das eine magische Welt erzeugt (Benutzeroberfläche).

Hardware: Die Hardware eines Computers (das, was beim Runterfallen klappert) ist die technisch-physikalische Grundlage für die Ausführung der mit dem Computer vereinbarten ***Spiele*** (der Software). Die Hardware umfaßt neben dem »Gehirn« des Computers (→ ***Prozessor***) auch Ein- und Ausgabegeräte (sogenannte Peripherie) wie Diskettenstationen, Drucker, ***Bildschirm, Tastatur***.

Ikone: Eigentlich ***kirchliches Heiligenbild***, ist häufig als Eindeutschung des englischen ***Icon*** zu hören: Bild(chen). Besser wäre der bei uns gebräuchliche Begriff ***Pictogramm*** (Bildsymbol, auf Tafeln an Flughäfen, Stadien, öffentlichen Gebäuden mit Hinweisen wie Telefon, Toilette, Fluchtweg etc. oft zu sehen).

Lisa: Computer von Apple, der am Anfang der ***populären Mediencomputer*** stand. Beruht auf Konzepten von Xerox (Star-Computer von PARC, Palo Alto Research Center) und eigenen Weiterentwicklungen und psychologischen Forschungen von Apple über Benutzeroberflächen (***magische Welt***).

Macintosh: Computer von Apple, preiswerteres Nachfolgemodell der ***Lisa***.

Magische Welt: Die magische Welt beruht darauf, daß der Computer auf seinem Bildschirm wie bei einem Trickfilm Abläufe mit scheinbar realen Gegenständen darstellt (dazu können auch Geräuscheffekte kommen). Im Gegensatz zum Film, bei dem der Zuschauer passiv bleibt, kann er in die Abläufe in der magischen Welt eingreifen, ja, der eigentliche Zweck der magischen Welt ist es, einen Dialog zwischen Computer und Mensch herzustellen.

Mediencomputer: Der Mediencomputer vereint die audio-visuellen Merkmale von Radio und Television (Fernsehen, Video) mit den Möglichkeiten eines Computers. Der Computer kann wie ein Synthesizer Geräusche, Töne und Klänge (Instrumentenimitationen) hervorbringen, eventuell sogar Sprachlaute (→ ***Magische Welt***). Dazu vermag er stehende und bewegte Grafik (Computertrickfilm) zu erzeugen. Zu erwarten ist in Zukunft die totale Integration von Fernsehen, Video (Kamera und Recorder), Faksimile-Drucker (zum Beispiel grafikfähiger Laserdrucker), Telefon, Post und Computer mit Auskunfts- und Bestellsystemen über Datennetze zu einem universalen Haushalts-Mediencomputer. Dieser Mediencomputer sorgt nicht nur für Unterhaltung, er wird auch mit der Bank des Besitzers, Kaufhäusern, öffentlichen Bibliotheken und so weiter in Verbindung stehen. Er erlaubt das Senden von grafischen Briefen (Handschrift, Zeichnungen) ebenso wie das Verlegen von Büroarbeit aller Art in das heimische Wohnzimmer.

Microsoft Windows: Fensterumgebung von Microsoft für den IBM-PC, mit Maussteuerung und Zeichenprogrammen. Kann – ähnlich GEM – konventionelle Programme einbinden.

Populärer Computer: Die klassische Unterscheidung zwischen Heimcomputer (preiswerter Hobbycomputer für den häuslichen Gebrauch, mit Betonung auf maschinennaher Programmierung und Computerspielen) und Büro- oder Personal Computer (Computer am Arbeitsplatz) beginnt wieder zu verschwimmen. Es gibt aber einen deutlichen Trend hin zu Computern, die – im Gegensatz zu Hackercomputern oder Bürocomputern – zu Konsumartikeln wie Fernsehgeräte oder Hi-Fi-Anlagen werden – wobei sie die Merkmale eines magischen Spielautomaten, einer Videoanlage, von TV-Telefon und Arbeitsgeräten immer mehr vereinen. Ein populärer Computer verfügt über Grafik (meist farbig und bewegt), Soundeffekte und so weiter, kann aber ebenso für die ernsthafte Programmierung oder für Anwendungen (Textverarbeitung, Büroaufgaben) eingesetzt werden.

Prozessor: Der Prozessor ist dasjenige Bauteil in einem Computer, das ihn von anderen Maschinen (insbesondere Automaten) unterscheidet. Der Prozessor ist ein elektronischer Baustein, der in elektronischer Form gespeicherte ***Spiel***regeln auszuführen vermag. Da die Spielregeln völlig beliebig sind, wird der Computer dadurch zu einer universalen Maschine, die im Prinzip für jeden beliebigen Zweck benutzt werden kann. Da der Prozessor in einem ***Mediencomputer*** außerordentlich viel zu tun hat, erhält er meist zu seiner Entlastung zusätzliche Coprozessoren (Hilfsprozessoren), die meistens für spezielle Zwecke (Grafik- oder Tonerzeugung) dienen. Dadurch wird ein Computer schneller und vielseitiger in seinen Ausdrucksmöglichkeiten → ***Magische Welt***.

RETURN: Eine besondere Taste auf der ***Tastatur***, mit der eine Eingabe abgeschlossen wird. Wichtig bei Anweisungen an den Computer, die in geschriebener Form erfolgen: Das Drücken der RETURN-Taste zeigt an, daß der Befehl vollständig ist und nun ausgeführt werden soll. Bei der Textverarbeitung dient die RETURN-Taste dazu, eine neue Zeile zu beginnen (ähnlich dem Wagenrücklauf auf der Schreibmaschine). Manchmal haben Computer noch eine eigene ENTER-Taste, die zum Abschluß der Eingabe von Zahlen dient, oder die RETURN-Taste heißt selbst ENTER.

Schreibtisch: Die graue (beim Amiga blaue) Grundfläche des Bildschirms wird in der ***Magischen Welt*** Schreibtisch (desktop) genannt – es ist sozusagen die Arbeitsfläche, auf der sich Karteikästen, Mülleimer, Fenster und – bei Amiga und Apple – auch einzelne Programme oder ***Dateien*** befinden können.

Software → Spiele

Spiele: Unter »Spiel« verstehen wir hier nicht »Spielerei«, auch nicht den mathematisch-spieltheoretischen Begriff. Ein Spiel ist eine gemeinsame Handlung, die auf der Übereinkunft der Teilnehmer über Regeln beruht, die anstelle der gewöhnlichen Verhaltensregeln treten oder sie ergänzen. Zum Beispiel darf man in einem Spiel nicht schummeln, kommt im Uhrzeigersinn an die Reihe und so weiter. Man kann Computerprogramme als Spiele auffassen, das heißt als Handlungen, die auf Regeln beruhen, über die ein Mensch mit einer Maschine (der Computer-***Hardware***) eine Vereinbarung trifft. Dann sind Programme (Software) Spielregeln. In unserem Zusammenhang sollten die Spielregeln einleuchtend und natürlich, nicht obskur und kompliziert sein.

Tastatur: Ein Eingabemedium, das mit den Fingern bedient wird. Es beruht auf dem Zeichensatz unseres geschriebenen Alphabetes und Zusatzzeichen sowie Tasten, die bestimmte Aktionen auslösen. Die Basis der Kommunikation mit dem Computer über die Tastatur ist die Schriftsprache. Gegensatz: gesprochene Sprache (Audio-Input), grafische Objekte (Gestalten).

Die Könner kommen

Sie wachsen den Eltern über den Kopf: Die 16-Bit-Computer übertreffen ihre 8-Bit-Ahnen bei weitem und sind preiswerter noch dazu. Rosa Zeiten also für die Freaks.

Wie bei den klassischen 8-Bit- oder den MS-DOS-Computern eignen sich auch 68000-Computer für verschiedene Aufgaben unterschiedlich gut. Allerdings sind die Einschränkungen nicht so gravierend. Jeder der hier beschriebenen Computer kann praktisch alles, was sowohl klassische Heim- wie auch Personal Computer leisten, nur eben in einem Gerät vereinigt. Wer mit deren Eigenschaften aber schon zufrieden ist, braucht keine 68000-Maschinen, denn nur wenn die zusätzlichen Fähigkeiten, wie zum Beispiel höhere Arbeitsgeschwindigkeit, größere Speicher, Multitasking und ähnliches tatsächlich genützt werden, ist eine Anschaffung sinnvoll. Wohlgemerkt: Vielleicht besteht der Nutzen »nur« darin, besonders schöne Entertainmentsoftware laufen zu lassen, wie etwa kreative Malprogramme. Auch der Einsatz als Freizeitinstrument ist nämlich Nutzen.

Und was viele an diesen Alleskönnern heute noch für »Übermotorisierung« halten, zum Beispiel Speichergrößen von 1 MByte und mehr, das kann morgen schon Mindeststandard sein. Die Entwicklung der letzten Jahre hat gezeigt, daß wachsende Speichergrößen und Arbeitsgeschwindigkeiten durch immer größer und komplizierter werdende Software schnell eingeholt wurden. Manche Anwendung wird durch solche Hardware erst möglich, ohne daß dies immer an der Programmoberfläche zu erkennen wäre. Allerdings – das hat die Erfahrung auch gezeigt – sinkt der Wille zum »ökonomischen« Umgang mit solchen Ressourcen bei den Programmierern mit der Verfügbarkeit. Man verbrät, was man hat.

Um in den vollen Genuß der zusätzlichen Fähigkeiten der 68000-Maschinen zu kommen, heißt es also bei diesen ebenso auf besondere Eignungen achten und gezielt wählen, wie bei den früheren Computergenerationen. Dabei wird Ihnen unsere Zusammenfassung der technischen Werte und Beschreibungen der gängigen 68000-Computer auf den folgenden Seiten eine große Hilfe sein. Jeder dieser Steckbriefe ist von einem Kenner des jeweiligen Computertyps geschrieben worden. Daraus erklärt sich der unterschiedliche Stil. Darauf bauen aber auch unsere Empfehlungen für Sie auf.

Salut für Mac

Kaum ein anderer Computer ist so kinderleicht zu handhaben wie der Macintosh. Über selbsterklärende Symbole und die Maus verständigt er sich mit seinem Benutzer. Dieses ausgefeilte Betriebssystem läuft auf einem Computer mit modernster Technologie. Praktische Hardware-Erweiterungen und ein Software-Angebot von über 200 Programmen machen aus dem Mac eine runde Sache.

Seit dem Frühling 1984 auf dem Markt, ist er bereits zum Veteran unter den Computern mit 68000-Prozessor geworden. Mit seinem Grafikbildschirm und dem grafischen Dialog mit dem Benutzer hat der Mac einen Trend gesetzt. Viele Anwender begrüßen das leicht erlernbare System. Sie haben mit dem Mac ein Werkzeug zum intuitiven Arbeiten gefunden. Für viele Soft- und Hardware-Entwickler wurde er zum Vorbild. Trotzdem blieb ihm der ganz große Erfolg, gemessen an Absatzzahlen, versagt. Weltweit wurden bisher etwa 500000 Macs verkauft. Rund 10000 Exemplare stehen davon in Deutschland. Anhänger der vom großen Konkurrenten IBM geprägten MS-DOS-Welt sehen im Mac auch heute noch gerne ein Spielzeug. In ihren Augen ist es nicht möglich, daß ein Computer leicht zu bedienen und trotzdem auch noch leistungsfähig ist. Doch Software wie das Datenbankprogramm Omnis oder die integrierte Software Excel machen aus dem kleinen Mac einen starken Computer. Auch Programmierer kommen voll auf ihre Kosten. Es gibt inzwischen Compiler und Interpreter für die gängigsten Programmiersprachen, mit denen sich so ziemlich jedes Programmierproblem lösen läßt.

Grafisch orientiertes Betriebssystem

Der Mac war zwar nicht der erste Computer, der mit einer Maus und einem grafisch orientierten Betriebssystem arbeitet. Jedoch erst mit dem Mac gelang Piktogrammen und dem Prinzip des Zeigens der Durchbruch.

Der Mac stellt alle wichtigen Dinge wie Disketten in seinem Laufwerk, auf den Disketten gespeicherte Dokumente und Programme, Systemdateien und seinen Papierkorb als kleine Symbole dar, die der Eindeutigkeit halber beschriftet sind. Jedes Symbol läßt sich mit der Maus auf dem Bildschirm bewegen. Rollt man die Maus auf der Tischfläche, bewegt sich synchron dazu ein Pfeil auf dem Bildschirm. Zeigt man mit dem Pfeil auf ein Symbol und drückt einmal kurz auf den Mausknopf, so färbt sich das Symbol schwarz. Dies bedeutet, daß es nun aktiviert ist; es wurde ausgewählt.

In der ersten Zeile des Mac-Bildschirms stehen die fünf Hauptmenüs: Apfel, Ablage, Bearbeiten, Inhalt und Spezial. Eine Funktion aus diesen Menüs kann man auswählen, indem man den Pfeil auf ein Hauptmenü bewegt, auf den Mausknopf drückt und ihn auch gedrückt hält. Dann klappt eine Liste von Funktionen unter der Menüzeile hervor und überlappt einen Teil des Bildschirms. Da die Funktionsliste nach unten herausgezogen wird, heißt diese Art von Menü Pull-Down-Menü.

Die Funktionsliste enthält sowohl grau als auch schwarz geschriebene Befehle. Auszuwählen und durchführbar sind lediglich die schwarz geschriebenen. Um eine Funktion auszuwählen, zieht man den Mauspfeil langsam nach unten auf eine der schwarz geschriebe-

nen Funktionen. Die Funktion, auf die der Pfeil zeigt, wird invers hervorgehoben. Läßt man den Mausknopf los, so wird die Funktion ausgeführt.

Nach diesem Prinzip funktioniert das gesamte Betriebssystem des Macintosh. Man zeigt auf etwas, wählt es aus, und der Mac beginnt zu arbeiten.

Die vier Grundrechenarten beherrscht der Mac-Taschenrechner. Seine Ergebnisse lassen sich in jedes Mac-Dokument einfügen.

Ein Wecker zeigt Datum und Uhrzeit. Er bleibt beim Ausschalten des Mac nicht stehen, da er von einer Batterie mit Strom versorgt wird.

Unverkennbares Design und Vorreiter: Macintosh

Überflüssige Programme, Dokumente oder Systemdateien kann man in den »Papierkorb« werfen. Bei herkömmlichen Betriebssystemen wäre hierbei von Löschen die Rede. Genauso wie beim Kopieren bewegt man einfach das entsprechende Symbol in den »Papierkorb« auf dem Bildschirm. Wenn man eine wichtige Datei wie zum Beispiel ein Programm wegwerfen will, verlangt der Mac eine Bestätigung. Vom Anwender erzeugte Dateien wandern jedoch ohne Rückfrage in den Papierkorb, der groß genug für mehrere Dateien ist.

Taschenrechner, Wecker und Notizblock

Wie aus einem echten Papierkorb kann man auch aus dem des Macintosh ein aus Versehen weggeworfenes Dokument wieder fischen. Dazu läßt man sich seinen Inhalt zeigen und bewegt das gesuchte Dokument heraus aus dem Abfall.

Unter dem Apfel-Menü bietet der Mac einige praktische Funktionen, die sowohl in Anwenderprogrammen als auch im Betriebssystem zur Verfügung stehen.

Über die Belegung der einzelnen Tasten informiert die Funktion Tastatur. Sie zeigt auch alle Doppelbelegungen in Verbindung mit der Option-, Shift-, Caps- oder Kommando-Taste.

In einem Kontrollfeld sind Systemvariablen wie die Lautstärke oder die Geschwindigkeit, mit der der Cursor blinkt, einzustellen.

Bilder und Texte, die man häufig benötigt, kann man in einem Album sammeln. Eine Seite des Albums läßt sich in jedes beliebige Dokument einfügen.

Für rasche Notizen hält der Mac einen Notizblock bereit. Die Notizen sind dabei unabhängig von dem Dokument, an dem gerade gearbeitet wird. Sie lassen sich genauso wie ein Blatt des Albums in ein Dokument einfügen. Umgekehrt ist es auch vorgesehen, einen Teil eines Dokuments an den Notizblock zu übertragen.

An einem Zahlenpuzzle kann man während einer Arbeitspause sein logisches Denkvermögen fördern.

Zugegeben, beim Mac steht die Software im Vordergrund. Doch auch seine Hardware weist viele interessante Eigenschaften auf.

Es gibt inzwischen drei Versionen des Mac. Zwei davon unterscheiden sich lediglich in der Kapazität des Arbeitsspeichers. Die ursprüngliche Variante verfügt über 128 KByte RAM. Diese RAM-Kapazität erwies sich jedoch sehr bald als nicht ausreichend. Daraufhin wurde der Fat Mac mit 512 KByte RAM entwickelt. Er ist inzwischen zum Standard-Mac geworden, da fast alle Programme den größeren Arbeitsspeicher benötigen. Der Speicher des 128-KByte-Mac kann auf 512 KByte ausgebaut werden.

Die dritte Version des Mac ist eigentlich eine Lisa. Die ehemalige Lisa 2/10 wurde umbenannt zu Macintosh XL. Der Arbeitsspeicher des Mac XL bietet eine Kapazität von 512 KByte und ist auf 1 MByte ausbaufähig. Zudem besitzt er eine 10 MByte Festplatte.

Keine Software ohne Hardware

Der Mac besteht aus drei Teilen: Systemeinheit, Tastatur und Maus. In der Systemeinheit sind ein Diskettenlaufwerk und der Bildschirm untergebracht. Das Herz des Mac ist die Digitalplatine, die auf dem Boden des Gehäuses installiert ist. Sie beherbergt den Mikroprozessor Motorola 68000, 128 KByte beziehungsweise 512 KByte RAM, 64 KByte ROM, den Disketten-Controller, die Chips zur Steuerung von Bildschirm und Maus sowie auch alle Schnittstellen.

Der Mikroprozessor Motorola 68000 arbeitet mit einer Taktfrequenz von 7,8336 MHz. Die Taktfrequenz ist entscheidend für die Rechengeschwindigkeit. Je höher die Taktfrequenz, desto höher die Rechengeschwindigkeit. Der Prozessor kommuniziert über einen 16-Bit-Datenbus beispielsweise mit den RAM- oder ROM-Chips. Intern arbeitet er in 32-Bit-Einheiten.

Wichtige Systemfunktionen sind in den 64 KByte der beiden ROM-Chips gespeichert. Etwa 500 in Assembler geschriebene Funktionen stehen dem Programmierer im ROM zur Verfügung. Dazu zählen kleine Programme zur Ansteuerung der Peripherie und zur Speicherverwaltung sowie auch Grafikroutinen, mit denen nahezu alle Ausgaben auf dem Bildschirm gestaltet werden.

Auf der Digitalplatine des Mac 128 KByte stecken 16 RAM-Chips in zwei Gruppen von je acht Chips. Jeder Chip hat einen Speicher von 64 KBit. Insgesamt ergibt sich dann ein Speicher von 128 KByte (64/8x16).

Der Mac 512 KByte ist mit derselben Anzahl an Chips ausgestattet. Die Chips sind jedoch von modernerer Bauart und speichern 256 KBit. Damit bie-

tet er eine Gesamtkapazität von 512 KByte (256/8x16).

Für Anwendungsprogramme steht jedoch nicht der gesamte Arbeitsspeicher zur Verfügung. Routinen des Betriebssystems und der Bildschirmspeicher belegen etwa 50 KByte.

Die Kapazität des Mac-Arbeitsspeichers dürfte jedoch kaum zum Problem werden. Anwender, die einen sehr großen Bedarf an RAM-Speicher haben, können ihn auf maximal 4 MByte erweitern. Damit sind die Grenzen für Programmierer sehr weit gesteckt.

Der Mac enthält sechs PAL-Chips (Programmable Array Logic), die genausoviel leisten wie 30 gewöhnliche integrierte Schaltungen. Damit wird viel Platz im Mac gespart. Der gesamte Mac besteht lediglich aus etwa 50 Chips.

Wenige Bausteine in einem Computer haben zudem den Vorteil, kaum Wärme zu entwickeln. Deshalb benötigt der Mac keinen ständig surrenden Ventilator, ohne den viele andere Computer sehr schnell lahm liegen würden.

Senkrecht zur Digitalplatine ist die Analogplatine montiert. Auf ihr sind das Netzteil, eine Uhr mit Batterieversorgung (4,5 Volt), ein Lautsprecher und Chips, die zur Ansteuerung der Bildröhre erforderlich sind, untergebracht. Die 9-Zoll-Bildröhre beansprucht den größten Teil der oberen Hälfte in der Systemeinheit.

Unter der Bildschirmröhre befindet sich das 3½-Zoll-Diskettenlaufwerk. Es passen 400 KByte auf eine Diskette. Der Mac beschreibt sie einseitig.

Ungewohnter Diskettenschutz

Es gibt keinen Knopf oder Riegel, um das Laufwerk zu öffnen oder zu schließen. Wenn eine Diskette einmal eingelegt ist, kann man sie nicht so ohne weiteres wieder entnehmen. Die Software des Mac erlaubt erst dann die Diskette zu entnehmen, wenn alle Schreibvorgänge beendet und alle Dateien geschlossen sind. Dadurch bleiben die Daten auf der Diskette vor vielen möglichen Schäden bewahrt.

Man kann an den Mac ein weiteres Laufwerk anschließen. Das externe Laufwerk ist für viele Anwendungen unbedingt erforderlich, da ansonsten laufend Disketten zu wechseln wären.

Der 9-Zoll-Bildschirm des Mac ist in 512x342 Punkte eingeteilt. Diese hohe Auflösung bietet der Mac auch bei der Darstellung von Text. Deshalb kann er Text in beliebigen Zeichensätzen und -größen ausgeben. Sehr angenehm für das Auge ist die Darstellung von schwarzen Zeichen auf weißem Hintergrund. Zudem ist das Bild sehr scharf und nahezu flimmerfrei. Farbe stellt er jedoch nicht dar.

Fast alle Mac-Programme nutzen die Maus. Das Prinzip, mit der Maus auf einen Befehl zu zeigen und ihn auch mit der Maus aufzurufen, erleichtert das Einarbeiten in neue Software ungemein. Auch Anfänger kommen so viel schneller mit neuen Programmen zurecht. Das Maus-Konzept ermutigt den Anwender, ein Programm im »Trial and Error-Verfahren« zu erforschen.

Die Tastatur ist mit dem deutschen Zeichensatz ausgerüstet. Sie hat keine Zehnertastatur und auch keine Funktionstasten. Sehr negativ fällt beim Arbeiten mit Textprogrammen oder auch Kalkulationsprogrammen auf, daß die Tastatur keine Pfeiltasten zur Cursor-Steuerung besitzt. Die Maus ist zwar bestens geeignet, den Cursor zu bewegen. Aber manchmal wären Cursorbewegungstasten praktischer, damit man die Hand nicht stets zwischen Tastatur und Maus hin und her bewegen muß.

An der Rückseite des Mac befinden sich fünf Anschlüsse. Einer davon ist für die Maus vorgesehen, ein anderer für das externe Diskettenlaufwerk und ein weiterer für einen externen Lautsprecher oder eine Stereoanlage. Zwei Anschlüsse sind serielle RS-422-A-Schnittstellen, über die ein Drucker und ein Modem oder andere Peripherie-Geräte angeschlossen werden.

Die beiden 9poligen seriellen Schnittstellen arbeiten mit einem Zilog 8530 SCC (Serial Communication Controller). Dadurch sind Übertragungsraten bis zu 230 KBit pro Sekunde möglich.

Neben jeder Buchse kennzeichnet ein Symbol, welches Gerät anschließbar ist.

Fest- und Wechselplatten für komfortables Arbeiten

Für den Mac gibt es inzwischen zahlreiche Fest- und Wechselplatten, deren Speicherkapazitäten zwischen 5 und 45 MByte liegen. Eine 5,5-MByte-Platte von Corvus kostet beispielsweise 8265 Mark. Der Preis für die 45-MByte-Corvus-Platte liegt bei 27 763 Mark. Die Festplatte HD 20 faßt 20 MByte und ist für 6840 Mark erhältlich.

Steckbrief Macintosh

Prozessor:	Motorola 68000
Taktfrequenz:	7,8336 MHz
RAM-Speicher:	128 oder 512 KByte erweiterbar auf maximal 4 MByte
ROM-Speicher:	64 KByte
Tongenerator:	vierstimmig
Uhr:	CMOS-Chip (4,5 Volt-Batterie)
Diskettenkapazität:	400 KByte, 1 eingebautes 3½-Zoll-Laufwerk
Bildschirm:	9-Zoll-Bildschirm mit 512x342 Punkten schwarz auf weiß
Tastatur:	deutsche Tastatur ohne Funktionstasten und Ziffernblock
Maus:	mechanische Maus
Schnittstellen:	zwei serielle Schnittstellen RS 232C/RS 422A, 230 Baud maximale Übertragungsrate
Größe (HxBxT):	34,4x24,6x27,6 cm
Gewicht:	7,5 kg

Preise in DM für Macintosh und Zubehör (inklusive Mehrwertsteuer)

Macintosh 128 KByte	7980,–
Macintosh 512 KByte	8949,–
Macintosh XL, 512 KByte	18500,–
Speichererweiterung von 128 auf 512 KByte	1140,–
Speichererweiterung von Mac XL auf 1 MByte	3780,–
Speichererweiterung von Mac 512 KByte auf 1 MByte	rund 4060,–
2 MByte	rund 5586,–
4 MByte	rund 8698,–
Macintosh Systempaket, bestehend aus: 1 Macintosh 512 KByte, 1 externen Laufwerk, 1 Imagewriter (15 Zoll), 1 10er-Tastatur, je 1 Macwrite, Macpaint, Macproject	12255,–
Externes Laufwerk	1695,–
20-MByte-Festplatte HD 20, eingebaut in Mac	6840,–
10er-Tastatur	495,–
Apple-Talk-Anschluß-Kit	250,–
Apple-Talk-Verbindungskabel 10 Meter	250,–
Imagewriter I 12 Zoll mit Anschlußkabel	1710,–
Imagewriter I 15 Zoll mit Anschlußkabel	2850,–
Imagewriter II (Farbe) mit Anschlußkabel	2850,–
Einzelblatteinzug für Imagewriter II	969,–
32 KByte Druckerpuffer für Imagewriter II	342,–
Laserwriter mit Toner	26790,–

Besonders platzsparend ist die Hyperdrive. Diese Festplatte kann in den Mac eingebaut werden. Die 10-MByte-Hyperdrive kostet 8637 Mark und die 20-MByte-Version 11 568 Mark.

Die hier aufgeführten Plattenlaufwerke stellen nur einen kleinen Teil des umfangreichen Angebots an Fest- und Wechselplatten dar.

Mac-Drucker

Der Imagewriter druckt jede Grafik und Schriftart, die man auf dem Bildschirm des Mac sehen kann. Es gibt mittlerweile zwei Versionen dieses Matrixdruckers. Der Imagewriter I druckt 120 Zeichen pro Sekunde im Textmodus. Für einen einseitigen Standardgrafikausdruck benötigt er 90 Sekunden und für die Ausgabe einer Grafik im hochauflösenden Modus 165 Sekunden. Der Imagewriter II druckt bedeutend schneller und vorallem auch in Farbe.

Der Ausdruck ist kaum noch von dem eines Typenraddruckers zu unterscheiden. Auch Grafiken druckt der Imagewriter zwar langsam, aber in hoher Auflösung. Er druckt sieben verschiedene Farben. Das Farbband ist in vier Bahnen mit den drei Grundfarben Rot, Gelb und Blau sowie auch Schwarz aufgeteilt. Drei weitere Farben entstehen durch Mischen der Grundfarben (Grün, Orange und Violett). Der Imagewriter II druckt somit alle Farben des Firmenemblems.

Beide Modelle des Imagewriter werden jeweils an der seriellen Schnittstelle mit dem Druckerzeichen angeschlossen.

Der Mac arbeitet jedoch auch mit vielen anderen Druckern mit seriellem Eingang zusammen. Für andere Modelle als den Imagewriter sind jedoch spezielle Druckertreiber erforderlich. Für alle gängigen Matrix- und Typenraddrucker wird jedoch die nötige Treibersoftware angeboten. Beispielsweise über die Mac Daisy Wheel Connection lassen sich verschiedene Typenraddrucker anschließen.

Das Tor zur Welt der 10 000 Programme für den IBM-PC ist Mac-Charlie für den Macintosh. Mac-Charlie erweitert den Mac um einen Intel-8088-Koprozessor, 256 oder 640 KByte RAM, ein 5¹/₂-Zoll-Laufwerk, zwei serielle Schnittstellen, den Anschluß für eine Erweiterungseinheit, einen Ziffernblock und Funktionstasten. Diese außergewöhnliche Ergänzung soll den Mac in die Lage versetzen, Disketten im Format des IBM-PC zu lesen und PC-Programme ohne Änderungen zu verarbeiten. Mac-Charlie soll für etwa 3300 Mark erhältlich sein.

Ohne Hardware-Erweiterung arbeitet der Mac auch mit den Betriebssystemen CP/M 68 und UCSD. Vor allem unter dem UCSD-System werden zahlreiche Compiler und Interpreter angeboten. Dazu zählen ANSI-Fortran 77, verschiedene C-Versionen, Lisp, Logo, Pascal, Forth, Modula, Assembler, APL und Basic.

IBM-PC und Mac in einem Gerät

Unter UCSD steht auch ein Basic-Compiler zur Verfügung. Die Möglichkeiten der Programmiersprachen, auf die Funktionen im ROM zuzugreifen, sind unterschiedlich. Ein Basic-Programmierer kann nur wenige ROM-Routinen nutzen, während der Pascal-Freak in der Lage ist, die meisten ROM-Funktionen in sein Programm einzubauen.

Auch Anwender von Standardsoftware treffen auf ein umfangreiches Angebot. Hierzulande gibt es derzeit mehr als 200 Programme für den Mac. Bewährte Programme wie Multiplan, DB Master, Pfs:File und Report, ein zu dBase III kompatibles Datenbankprogramm, MS-Chart und MS-Word wurden an das Mac-Betriebssystem angepaßt. Entwickler haben jedoch auch jede Menge neue Software speziell für den Mac geschrieben. Dazu gehören die integrierten Programme Jazz, Excel, Ensemble und Quartett sowie auch grafische Datenbanken wie File-vision.

Ein reifes System Eine reife Leistung

Der Mac hat jedoch vor allem mit seinen Grafikprogrammen Macpaint und Macdraw Freunde gewonnen. Bildersammlungen, weitere Zeichensätze und Bibliotheken mit zusätzlichen Grafikfunktionen erweitern das Grafikspektrum des Mac.

Seine Kinderkrankheiten hat der Mac gut überstanden. Es gibt ein zweites Laufwerk, Fest- und Wechselplatten und sogar eine RAM-Erweiterung. Über geeignete Treibersoftware lassen sich fast alle Drucker ansteuern. Auch das seit langem erwartete Netzwerk Apple-Talk ist nun endlich da. Trotzdem das Mac-Betriebssystem vorbildlich ist, sind die Betriebssysteme UCSD und CP/M auf dem Mac zu begrüßen. Über Mac-Charlie läuft sogar MS-DOS auf dem Mac. Das umfangreiche Software-Angebot macht den Mac sicherlich für viele zu einem attraktiven Computer.

(Silvia Gutschmidt/lg)

Kosinus von GUBA & ULLY

Ein Profi nicht nur für zu Hause

Viele Jahre verband den Namen Atari weder etwas mit Heim- noch mit Personal Computern, dann begann Atari seine wechselvolle Geschichte mit der Produktion von Videospielen. Diese Vergangenheit wollte sie selbst dann nicht verleugnen, als die Firma ihre ersten Heimcomputer auf den Markt brachte. Wichtiger Bestandteil dieser Geräte war der Einschubschacht für Spiel-Module. Diese Tatsache wurde in der Werbung immer noch groß herausgestellt, als die Heimcomputer längst zu neuen Ufern aufgebrochen waren und sich mehr und mehr ernsthafteren Aufgaben zuwandten. So war es denn auch nicht verwunderlich, daß die schon damals sicherlich leistungsfähigen Atari-Heimcomputer weiterhin mit dem etwas zweifelhaften Ruf des Spielcomputers behaftet waren. Im großen Preiskrieg der Heimcomputeranbieter 1982/83 siegte dann der Computer-Guru Jack Tramiel, damals an der Spitze des Commodore-Konzerns, mit Commodores C 64 endgültig, und manövrierte damit Atari an den Rand des Ruins. Derselbe Jack Tramiel trennte sich nur wenig später, 1984, im Streit von Commodore und kaufte zur Überraschung aller für wenig Geld die Scherben von Atari, die er selbst kurz vorher verursacht hatte. Nach und nach folgte ihm fast das ganze Erfolgsteam, das den C 64 zu einem Standard in der Heimcomputerszene gemacht hatte. Was war da zu erwarten? Bekanntlich ist der große Jack, wie er oft liebevoll bezeichnet wird, immer für eine Sensation gut. Aber ob Sensationen auf der Basis einer so maroden Firma wie damals Atari gedeihen könnten, schien vielen selbsternannten Computerweisen doch sehr fraglich.

Phönix aus der Asche

Zuerst auf der C.E.S. in Las Vegas im Januar 1985 und dann auf der Hannover-Messe 1985, platzte die Bombe. Wie der sprichwörtliche Phönix aus der Asche erhob sich eine neue Firma Atari unter Jack Tramiel aus den Scherben des Videospielproduzenten. Sie präsentierte nach beinahe unglaublich kurzer Entwicklungszeit einen wahren Supercomputer, den Atari 520 ST. Zum Preis einer gut ausgebauten Heimcomputeranlage wurden zukunftweisende Eigenschaften versprochen, die auch weitverbreitete und teure Personal Computer mit nur drei Großbuchstaben im Firmenzeichen deutlich überflügeln könnten. Erdreistete sich etwa das Produkt einer Spielcomputerfirma, in Domänen der großen blauen Mutter IBM einzubrechen?

Billiger im Paket

Doch in unserer schnellebigen Zeit ist auch das Jahr 1985 schon Geschichte. Atari beließ es nicht bei sensationellen Ankündigungen, sondern lieferte seinen ST entgegen aller Skepsis Mitte 1985, wie angekündigt, als Paket zum Preis von 2998 Mark aus. Dieses Paket beinhaltete einen Computer mit 512 KByte RAM, ein Diskettenlaufwerk mit etwa 360 KByte nutzbarer Speicherkapazität, einen phantastischen Monochrom-Monitor und eine Maus als Bedienungselement. An Software waren das Betriebssystem TOS, die Benutzeroberfläche GEM und die Computersprachen Basic und Logo jeweils auf 3½-Zoll-Disketten beigelegt. Zur Systems, Ende Oktober 1985, erfolgte dann der bislang letzte Paukenschlag. Atari präsentierte den 520 ST+ mit einem RAM-Speicher von einem Megabyte und den 260 ST mit 512 KByte. Im Gegensatz zum 520 ST+ wird der 260 ST nicht mehr im Paket angeboten, sondern als einzelne Einheit ohne Laufwerk, Monitor und Maus verkauft.

Die derzeit von den Atari-Systemhändlern lieferbaren Komponenten und ihre Listenpreise sind in der Tabelle aufgeführt. Die Paketpreise für den 520 ST+ mit Laufwerk, Monitor und Maus sind gut 400 Mark niedriger, als die Summe der Einzelpreise. An dieser Stelle ist zu bemerken, daß zu einem funktionsfähigen ST-System allerdings unbedingt eine Diskettenstation und selbstverständlich ein Monitor gehört. Zur Not kann der Computer auch ohne Maus mit den Cursor-Tasten bedient werden, richtig bequem wird es aber erst mit der Maus. Damit beträgt der »Mitgliedsbeitrag« zur Aufnahme in die ST-Gemeinde 2642 Mark. Bei genauem Studium von Fachzeitschriften stößt man jedoch schon auf einige Selbsthilfelösungen, die den etwas kostspieligen Einstieg in die Computerzukunft auf dem Atari ST-System um einige 100 Mark senken. Bleiben wir jedoch bei den Originalkomponenten von Atari. Was bekommt man für sein Geld? Die beiden Computer 260 ST und 520 ST+ unterscheiden sich ausschließlich in der Größe des eingebauten RAM-Speichers. Der 260 ST ist aber jederzeit mit etwas Lötarbeit (aber auch Garantieverlust) auf die Speichergröße des 520 ST+ ausbaubar. Der Preis für die benötigten 16 RAM-Bausteine beträgt derzeit etwa 130 Mark. Der 512-KByte-Speicher im 260 ST reicht für die augenblicklich verfügbare Anwendersoftware aus. Das Betriebssystem, das noch von der Diskette ins RAM geladen werden muß, und der Bildschirmspeicher verbrauchen aber schon etwa 260 KByte. Da große Anwenderprogramme etwa 150 KByte belegen, bleiben nur noch knapp 100 KByte für Daten übrig. Die ROM-Version des Betriebssystems soll aber in der ersten Hälfte dieses Jahres auf dem Markt sein und kann leicht nachgerüstet werden. Für einen Preis von 100 bis 120 Mark stehen dann zusätzliche 198-KByte-Speicher bereit. Beim 520 ST+ sind mit oder ohne ROM naturgemäß jeweils 512 KByte mehr ansprechbar. Nach der Devise, daß man nie genug Speicherplatz haben kann, ist ein freier Speicher von ungefähr 800 KByte für Daten sicherlich nicht zuviel. Wer weiß denn schon heute, ob der Phantasie der Programmentwickler nicht bald Programme von viel größerem Umfang als 150 KByte entspringen werden.

Das Superhirn

Ein großer Speicher allein macht noch keinen großen Computer. Auch die anderen Bausteine sollten vom Feinsten sein. Atari hat sich trotz des geringen Preises wahrlich nicht lumpen lassen. Als Gehirn des Atari ST wurde einer der fortschrittlichsten und leistungsfähigsten Mikroprozessoren gewählt, die der Markt zu bieten hat: Es ist der MC 68000 von Motorola, der im ST mit einer Frequenz von acht Megahertz getaktet wird. Der 68000 besitzt einen Datenbus mit einer Breite von 16 Bit. Intern verarbeitet er aber die Daten in 32-Bit-Registern, daher hat der ST auch seinen Namen »Sixteen/Thirty-two«, auf deutsch sechszehn/zweiunddreißig. Der Adreßbus mit 24 Bit ist in der Lage, bis zu 16 Megabyte Speicher direkt zu adressieren. Bei der Arbeit unterstützen den Mikroprozessor einige Peripheriebausteine, die zum Teil Standardelemente sind, zum anderen Teil aber Eigenentwicklungen von Atari darstellen. Zwei sollen hier besonders erwähnt werden.

Verantwortlich für die Speicherverwaltung ist die MMU (Memory Managing Unit), ein Baustein, der auf einfachste Weise Speichererweiterungen bis auf vier Megabyte zuläßt. Vorausset-

zung dazu sind allerdings die sogenannten Megachips mit einer Speicherkapazität von einer Million Bit pro Baustein, die heute erst in kleinen Stückzahlen zu astronomischen Preisen erhältlich sind. Dies kann man als sicheres Indiz dafür ansehen, daß Atari den ST nicht als kurzlebiges Produkt geplant hat.

Ebenfalls in die Zukunft weist ein anderer Baustein nach Entwürfen der eine ROM-Disk auf der Basis von Laserabtastbaren optischen Speicherplatten mit extremer Speicherkapazität und Zugriffsgeschwindigkeit, und ein Festplattenlaufwerk, das im Gegensatz zur ROM-Disk in absehbarer Zeit ausgeliefert werden kann. Bei einer Speicherkapazität von 20 Megabyte ist ein bisher kaum für möglich gehaltener Preis von unter 2000 Mark im Gespräch. Ein funktionsfähiger Prototyp dieses Fest- der direkt vom DMA-Chip bedient wird. Die Datenübertragungsrate beträgt etwa 250 KBit pro Sekunde.

Eine serielle Standardschnittstelle nach der RS232-Norm erlaubt den Anschluß entsprechend ausgestatteter Peripheriegeräte, wie zum Beispiel eines Akustikkopplers oder eines Modems für Datenfernübertragung. Auch Drucker mit seriellem Anschluß können hier mit Daten versorgt werden.

Zentraleinheit und Tastatur sind beim Atari ST in einem Gehause zusammengefaßt

ST-Entwickler, der DMA-Chip. »DMA« steht für Direct Memory Access. Als eine Art Hilfsprozessor kann er selbständig auf den gesamten Speicher zugreifen (sofern sein Chef, der Zentralprozessor, es zuläßt) und auf besonders schnelle Weise den Datenaustausch mit Peripherieeinheiten erledigen. DMA-Bausteine finden sich im allgemeinen nur in Personal Computern der höheren Preisklassen. Beim Atari ST fällt zunächst einmal nur die Diskettenstation in den Aufgabenbereich des DMA-Chips. Dies läuft aber eher nebenbei und lastet ihn in keiner Weise aus. Seine wahre Leistungsfähigkeit zeigt der Chip erst, wenn Peripherieeinheiten am sogenannten Harddisk-Port des ST angeschlossen werden. Der Name »Harddisk-Port« ist dabei ein wenig irreführend. Da alle wichtigen Ausgangsleitungen des DMA-Chip an diesen Port geführt sind, wäre DMA-Port die treffendere Bezeichnung. Das Betriebssystem TOS ist darauf vorbereitet, über den DMA-Port bis zu zehn intelligente Peripheriegeräte anzusprechen. Atari hat bisher zwei Einheiten vorgestellt, plattenlaufwerks war bereits auf der Systems 1985 zu bewundern.

Der DMA-Port ist selbstverständlich nicht das einzige Tor zur Außenwelt. Im Gegenteil, der Atari ST ist wie kaum ein anderer seiner Zunft mit Ports großzügig bestückt. Die serielle Laufwerk-Schnittstelle ermöglicht den Anschluß von bis zu zwei Atari-Diskettenstationen SF 354 oder SF 314. Beide Laufwerke benutzen 3½-Zoll-Disketten.

Anschlußfreudig

Die SF 354 beschreibt die Disketten einseitig mit 80 Spuren zu je 9 Sektoren. Die Speicherkapazität einer formatierten Diskette beträgt dabei knapp 360 KByte. Die SF 314 benutzt dagegen beide Seiten der Diskette und kommt damit auf die doppelte Speicherkapazität. Sie kann aber auch wie die SF 354 arbeiten und einseitig formatierte Disketten lesen und beschreiben. Die Laufwerke werden nicht unmittelbar vom DMA-Chip, sondern von einem Diskcontroller-Baustein angesteuert, Wer einen Drucker mit Parallelschnittstelle bevorzugt, wird auch nicht enttäuscht, da zusätzlich ein sogenannter Centronics-Port vorhanden ist. Übrigens ist dieser Anschluß mit ein paar Programmiertricks nicht nur als Ausgabe-, sondern auch als paralleler Eingabeport einzusetzen.

Die Monitorbuchse im äußeren Gewande einer DIN-Buchse sieht auf den ersten Blick nicht besonders bedeutend aus, enthält aber nicht weniger als 13 Kontakte. Hier werden der Monochrom- oder der RGB-Farbmonitor angeschlossen. Durch eine geschickte Beschaltung dieser Buchse erkennt das Betriebssystem des ST, welcher Monitor angeschlossen ist, und paßt die Bildauflösung selbständig an das jeweilige Gerät an. Auf dem wirklich phantastischen Monochrom-Monitor SM 124 erscheinen Text und Grafik, beliebig gemischt, zweifarbig (schwarz und weiß) in 640 x 400 Punkte aufgelöst. Texte werden selbstverständlich mit 80 Zeichen pro Zeile dargestellt. Dabei läßt die Bildschärfe selbst in den Bildecken nichts zu wünschen übrig,

das Bild ist absolut flimmerfrei. Ohne die geringste Übertreibung kann man diesen Monitor als eines der Glanzstücke des ST-Systems bezeichnen.

Der Farbmonitor SC 1224 (oder über ein Spezialkabel auch andere RGB-Monitore) wird über den RGB-Ausgang in der 13poligen Monitorbuchse betrieben. Dabei beträgt die Auflösung 640 x 200 Punkte in vier Farben oder 320 x 200 Punkte in 16 Farben. Die jeweiligen Farben können aus einer Palette von 512 verschiedenen Farbtönen ausgewählt werden.

Ein außergewöhnliches Attribut des Atari ST ist eine bei Mikrocomputern noch sehr selten zu findende Schnittstelle, der MIDI-Anschluß. Er dient der Steuerung von entsprechend ausgestatteten elektronischen Musikinstrumenten. Die hohe Übertragungsrate dieser bidirektionalen Schnittstelle läßt aber sicherlich noch einige andere Anwendungen zu. Als Beispiel dafür bietet sich die Verkettung mehrerer STs zu einem Netzwerk an.

Flottes Interface: MIDI

Die Software für eine solche Vernetzung mehrerer Computer könnte sich in einem ROM-Modul befinden, das in den ROM-Port auf der linken Seite des ST eingesteckt wird. Da die Read/Write-Leitung des Prozessors nicht auf den ROM-Port gelegt wurde, ist keine Verwendung als Systembus möglich. Es handelt sich um einen reinen Eingabeport für ROM-residente Programme. Die Modulprogramme mit einer Maximallänge von 128 KByte sind bei entsprechender Programmierung in der Lage, das Betriebssystem wesentlich zu beeinflussen. Dies kann bis zur völligen Übernahme des Systems durch das Modul im ROM-Port gehen.

Die Schaltzentrale

Es handelt sich beim Atari ST um einen sogenannten Tastaturcomputer. Tastatur und Zentraleinheit befinden sich also in einem gemeinsamen Gehäuse. Alle anderen Bestandteile des ST-Systems werden mit einer Kabelverbindung an dieses Gehäuse angeschlossen. Die Tastatur gliedert sich in vier getrennte Tastenfelder. Über der Haupttastatur, deren Tastenbelegung im wesentlichen der deutschen DIN-Norm entspricht, befindet sich eine Reihe mit zehn Funktionstasten. Diese können von Anwenderprogrammen mit bestimmten Programmsteuerfunktionen belegt werden. Rechts neben der Haupttastatur sind in einem Block die Cursorsteuertasten und ein abgesetzter Ziffernblock angebracht. Die Tasten lassen sich leichtgängig und griffig betätigen. Höhe und Neigung der Tastatur sind ergonomisch gut gestaltet, und ermöglichen ermüdungsfreies Arbeiten auch über einen längeren Zeitraum. Das Fehlen eines Druckpunktes ist etwas gewöhnungsbedürftig; ein abschaltbarer Tastaturklick macht aber diesen kleinen Nachteil fast wieder wett.

Mausmanager

Als scheinbar letztes Relikt aus der Spielcomputer-Vergangenheit befinden sich auf der rechten Seite des Computers zwei Joystick-Ports. An einen dieser beiden Ports wird die Maus als wichtigstes Eingabegerät des ST angeschlossen. Mit Hilfe der Atari-Maus, einem kleinen Kästchen mit zwei Tasten und einer Rollkugel auf der Unterseite, läßt sich der Atari ST auf sehr komfortable Weise bedienen. Hardware mag noch so fortschrittlich konzipiert sein, ohne ein ebenso fortschrittliches Betriebssystem ist sie jedoch völlig wertlos. Zum Glück hat Atari nicht versucht, ein absolut neues und damit absolut unbekanntes Betriebssystem zu kreieren, sondern ein in der Grundstruktur wohlbekanntes und gut dokumentiertes weiterentwickelt. Den Grundstock des ST-Betriebssystem TOS (Tramiel Operating System) bildet eine CP/M-Version für den MC 68000, das CP/M-68 von Digital Research. Ataris Softwareentwickler haben es noch um einige wichtige Funktionen bereichert, die normalerweise nur MS-DOS von Microsoft bietet. Hier ist in erster Linie die Diskettenverwaltung mit der Fähigkeit zur Anlage von Subdirectories zu nennen, die ein effektives Arbeiten mit großen Massenspeichern, wie zum Beispiel Festplattenlaufwerken, erleichtert. TOS ist damit weder zu CP/M-68 noch zu MS-DOS voll kompatibel. Die nahe Verwandtschaft zu CP/M-68 läßt aber eine Anpassung von Programmen für dieses Betriebssystem ohne größere Schwierigkeiten zu. Eine genauere Beschreibung finden Sie in dem Artikel »Kauderwelsch« in dieser Ausgabe.

CP/M-Systeme bieten zwar viele Funktionen, gelten aber als schwer bedienbar. Dieser weit verbreiteten Meinung ist Digital Research mit der Entwicklung der grafisch orientierten Benutzeroberfläche GEM entgegengetreten. Der mit diesem System verbundene Bedienungskomfort paßte offensichtlich hervorragend in das Konzept für das ST-System. Atari hat eine an TOS angepaßte Version von GEM entwickelt und in die Software-Grundausstattung des ST übernommen.

Bei der Entwicklung von GEM hat man sich einer Idee bedient, die den Anwender nicht mehr dazu nötigt, bei der Kommunikation mit seinem Computer auf dessen zugegebenermaßen primitive Sprachebene herabzusteigen.

GEM, wie »Edelstein«

Die grafische Benutzeroberfläche GEM simuliert auf dem Bildschirm einen für die Datenverarbeitung typischen Arbeitsbereich. Mit nur wenig Vorstellungskraft sieht man die Arbeitsfläche eines Schreibtisches, Aktenschubladen und einen Papierkorb vor sich. Ein kleiner Pfeil symbolisiert die arbeitende Hand des Menschen. Diese »Hand« kann mit Hilfe der Maus über die Arbeitsfläche bewegt werden und die Aktenschubladen öffnen, Blätter herausziehen, diese Blätter nach der Bearbeitung wieder in die Schubladen zurücklegen oder sie in den Papierkorb werfen. Der Atari ST setzt diese Operationen in Abläufe um, die ohne solch eine grafische Benutzeroberfläche bisher nur durch viele, häufig schwer lernbare Befehle, in »Computerchinesisch« auslösbar waren. Hinter den Aktenschubladen verbergen sich nämlich die Diskettenlaufwerke, und die Blätter ent-

Produkt	Preis
Atari 520 ST+ mit 1 MByte RAM, TOS, GEM, Basic, Logo	1698 Mark
Atari 260 ST mit 512 KByte RAM, TOS, GEM, Basic, Logo	1298 Mark
Diskettenlaufwerk SF 354, 360 KByte (formatiert)	598 Mark
Diskettenlaufwerk SF 314, 720 KByte (formatiert)	798 Mark
Monochrommonitor SM 124, Schwarz/Weiß	598 Mark
Farbmonitor SC 1224	1298 Mark
Maus für 520 ST+ oder 260 ST	148 Mark
Komplettsystem 520 ST+ bestehend aus: Atari 520 ST+, SF 354, SM 124, Maus, TOS, GEM, Basic, Logo	2998 Mark

Das Atari ST-System und sein Preis

halten nichts anderes als die auf den Disketten gespeicherten Dateien und deren Daten. Die Diskettendateien werden ebenfalls durch grafische Symbole dargestellt und können mit Hilfe der Maus manipuliert werden.

Die dressierte Maus

GEM ist aber nicht nur ein hervorragendes Hilfsmittel zur Bedienung des ST-Systems, sondern bietet auch die Möglichkeit, komplexe Anwenderprogramme ebenso benutzerfreundlich zu gestalten. Es werden bereits einige außerordentlich leistungsfähige Programme für die verschiedensten Anwendungsbereiche angeboten. Als zwei Beispiele unter vielen seien an dieser Stelle nur Textverarbeitung und Tabellenkalkulation genannt. Naturgemäß kann das Programmangebot für einen erst relativ kurze Zeit auf dem Markt befindlichen Computer wie den Atari ST noch nicht so vielfältig sein, wie für seine schon länger eingeführten Konkurrenten. Die stürmische Entwicklung des Programmangebotes in der zweiten Hälfte des Jahres 1985 läßt jedoch hochgesteckte Erwartungen in dieser Hinsicht nicht unrealistisch erscheinen.

Das Nest des Phönix

Von der Erfüllung dieser Erwartungen hängt sicherlich auch die endgültige Beantwortung der anfänglich gestellten Frage nach der Einordnung des Atari ST ab. Welcher der beiden beinahe klassischen Abteilungen Heim- oder Personal Computer ist der ST zuzuordnen? Vieles in diesem ersten Jahr seit seiner Markteinführung erinnert frappant an die Einführung des C 64 von Commodore. Der Personenkreis der ST-Besitzer war bis zum Winter 1985 vergleichbar mit den damaligen C64-Käufern, wenn auch der Altersdurchschnitt vielleicht etwas gestiegen ist. Nachdem der Atari ST anfangs zu den echten Freak-Computern zählte, hat sich dies jetzt bereits deutlich geändert. Die Tatsache, daß der Charakter der zuerst angebotenen Programme ganz eindeutig professioneller Natur war, hat dazu beigetragen, daß der ST immer mehr in den Bereich vorstößt, den seine Väter ihm wohl von Anfang an zugedacht haben.

Atari bezeichnet den ST als Personal Computer. Von seiner potentiellen Leistungsfähigkeit her gehört er auch zweifellos in eine Spitzenposition unter den Personal Computern. Sein äußerers Erscheinungsbild als Tastaturcomputer mag die Akzeptanz bei einigen Anwendern, zum Beispiel im Bürobereich, sehr wohl ein wenig vermindern. Das augenblicklich noch konkurrenzlos gute Preis/Leistungs-Verhältnis dieses technischen Wunderwerks, im Zusammenwirken mit leicht bedienbarer Software von hohem Leistungsstandard, wird dieses Manko jedoch in vielen Fällen ohne Schwierigkeiten überwinden können. Mehr noch, gerade der sensationell niedrige Preis wird dem Atari ST in breiter Front professionelle Anwendungsbereiche erschließen. Seine weitaus teureren und nicht einmal leistungsfähigeren Konkurrenten können davon nur träumen. Mit dem Atari ST wird es zum ersten Mal möglich, professionelle Computertechnik mit vertretbarem finanziellen Aufwand auch auf dem heimischen Schreibtisch zu nutzen. Vielleicht wird der Atari ST einmal als Urahn einer neuen Computergeneration in die Annalen eingehen, nämlich als der erste Privat-Computer. Übrigens, spielen kann man mit ihm auch...

(W. Fastenrath/aw)

Faszination Amiga

Mit vielen Vorschuß-Lorbeeren bedacht: der Amiga

Programmierer schweben im Delirium, Anwender werfen ihren alten PC weg, Spielefreaks gehen nicht mehr in die Spielhallen. Alle bewundern den Amiga, auch die weltweite Presse berichtet in höchster Verzückung. Warum?

Für viele ist der Amiga nicht nur ein Computer, sondern ein Phänomen. Obwohl wesentlich schwieriger zu programmieren als zum Beispiel ein Apple II oder ein C 64, sprechen eine ganze Menge Programmierer vom Amiga als ihrem Traumcomputer.

Englische Zeitschriften berichten, daß Jeff Minter, seines Zeichens Groß-

britanniens ausgeflipptester Computerfreak, nach einer Amiga-Vorführung nicht mehr so recht ansprechbar gewesen sei. Popular Computing Weekly hat diesen Zustand mit der Bezeichnung »Delirium« umschrieben.

Doch nicht nur der Programmierer, sondern auch der Endanwender könnte seinen Traum mit dem Amiga gefunden haben. Die Fähigkeiten des Gerätes haben einiges für sich, was den Business-Bereich betrifft. Im Grunde genommen kann der Amiga das, was der Macintosh und der IBM zusammen können. Das wird auch bewiesen, und nicht zu knapp. Wer Programme vom Macintosh umschreiben will, kann auf die »Mac Library« für das Amiga-Entwicklungssystem (Lattice C) zurückgreifen. Was der Amiga nicht schon an Mac-Fähigkeiten im Betriebssystem hatte, kann man nun dazubinden. Noch weiter wird die Sache mit dem IBM-Emulator getrieben. Wer den Amiga und den Emulator hat, darf seinen PC jetzt vergessen, denn die Kompatibilität ist verblüffend groß. Für Business-Software ist also in genügender Menge gesorgt.

Die Spiel-Maschine

Aber auch die Spielefreaks sind begeistert. Kein Wunder, konnte man diese Grafikfähigkeiten doch bisher nur in der Spielhalle bewundern. Gelockt wird der Spielhallenkenner schon durch Marble Madness, ein Spiel mit einmaligem Sound und toller Grafik. Die Werbung der Firma Electronic Arts bestätigt diesen Eindruck mit den Worten, das Spiel würde am Computer so gut werden wie der Automat. Die Fähigkeiten des Amiga lassen dies auch ohne weiteres zu; die ersten Grafikdemos, die man dazu zu sehen bekam, zeigten ein Bild, von dem man nicht wußte, ob man nun den Automaten oder den Computer vor sich hat. Jeder Spieler träumt nur davon, so etwas auch zu Hause stehen zu haben.

Die jungen Computerfans, träumen schon aus dem einfachen Grunde vom Amiga, weil sie ihn sich nicht leisten können. Beispiel: Auf der Commodore-Messe 1985 in Frankfurt wurde der Amiga hinter den Kulissen gezeigt. Eins wurde aber vergessen – die Schlitze in der Wand. Unmengen junger Computerenthusiasten drängten sich an dieser Wand, um durch die Schlitze die »Amiga-Peepshow« zu bewundern. Ein Computer wird zum Mythos in der Computerszene – etwa vergleichbar mit den Beatles aus der Popszene. Kaum ein Computer hatte soviel Popularität in der Presse und der Öffentlichkeit.

Durch die Faszination dieses Computers hat die amerikanische Presse glatt vergessen, den Konkurrenten (Atari ST) überhaupt zu erwähnen, nimmt man mal die Atari-treuen Zeitschriften aus. Der Atari ST ist zwar die preisgünstigere Alternative, aber bei dem, was der Amiga bringt, vergißt man leicht den Rest.

Faszination durch Technik

Warum der Amiga nun von allen so toll gefunden wird, kann man eigentlich schwer sagen, ohne sich in technischen Details zu verlieren. Tests und Vorstellungen wurden schon zur Genüge veröffentlicht, deswegen nachfolgend hier eine kurze Zusammenfassung der wichtigsten technischen Daten:

– 68000 als CPU: in der 16/32-Bit-Technologie der Standard. Ein Prozessor, der einiges mehr an Leistung bringt als so mancher andere.

– Anwenderfreundliche Benutzeroberfläche und Maus-Steuerung, mit Window-Technik ideal für den Erstanwender, für Benutzer der »alten Schule«, aber auch als »Command Line Interpreter« (CLI) für ganz normales Arbeiten über die Tastatur.

– Das Betriebssystem AmigaDOS erlaubt Multitasking: Mehrere Anwendungen können gleichzeitig benutzt werden.

– Außergewöhnliche Grafikfähigkeiten: Auflösung 320x200 oder 640x400, 4096 Farben, davon 32 gleichzeitig, im Hold&Modify-Modus alle 4096 Farben gleichzeitig, aber unter Einschränkungen.

8 Hardwaresprites, 16 Pixel breit, beliebig hoch. Jedes der 8 Sprites kann beliebig oft auf dem Bildschirm dargestellt werden. Zusätzlich Softwaresprites (Blitter Objects und Gels).

Faszinierende Demo-Programme

Durch einfache Registeränderung beliebig scrollbare »Playfields« (Hintergrundgrafiken, die größer als der Bildschirm sein können), auch mehrere Playfields gleichzeitig, transparent übereinander etc.

– Naturgetreuer Sound, weil Töne digitalisiert werden können.

– Sprachausgabe, die viele Variationen zuläßt. Eine männliche oder weibliche Stimme, mit aggressiver, normaler oder einschläfernder Stimmlage.

– DMA-Technik: Drei Custom Chips nehmen dem 68000-Prozessor viel Grafik-, Sound- und I/O-Arbeit ab und können gleichzeitig arbeiten. Die DMA-Chips sind in den meisten Fällen auch schneller als entsprechende Maschinenroutinen in 68000-Assembler (der ohnehin schon schnell ist)

– und, und, und…

Der Amiga ist ein Computer, der nicht für eine bestimmte Anwendergruppe gedacht ist, sondern jedem etwas bietet. Nicht zuletzt dadurch bestaunt jeder das Gerät, zumal es bisher noch nichts Derartiges gab.

Doch hinter der Faszination dieses Computers steckt noch mehr, nämlich Geld. Es wurde viel ausgegeben, um Demonstrationssoftware zu schreiben, die den Computerkäufer und nicht zuletzt die Presse beeindrucken sollen. So etwas wie den springenden Ball, der nun schon zum Amiga-Wahrzeichen geworden ist, gibt es inzwischen auch auf dem Atari ST, dem Macintosh und anderen Computern. Für Demografik wurden Künstler bei Amiga angestellt. Man arbeitet mit Softwarefirmen zusammen. Atari hätte das alles auch haben können – wenn genügend Geld in Demos gesteckt worden wäre, würde der Atari wohl auch etwas beeindruckender sein. Commodore hat sich das gut überlegt. Denn Kunden holt man sich nicht nur mit technischen Details, sondern ganz einfach mit Faszination durch Demonstrationen und Show. Auf den Seminaren für Softwareentwickler wurde denn auch mehr Show geboten als Technik. Was haben Smarties als erster Gang des Abendessens oder eine Stoffpuppe des rosaroten Panthers mit dem Amiga zu tun?

Amiga-Show

Eigentlich nichts. Aber wer die Softwareentwickler-»Show« (Konferenz ist wohl der falsche Ausdruck) in England miterlebt hat, wird sich sicherlich zu beidem den Anblick des Amiga ins Unterbewußtsein rufen. Die sagenhafte Grafik des Amiga auf einer mehr als 50 Quadratmeter großen Leinwand zu sehen, den Stereosound unterstützt durch die leistungsstarke Verstärkeranlage des Hotels zu hören, hat auch die kritischen Betrachter sehr überzeugt und einen bleibenden Eindruck hinterlassen. Das ist Faszination, wie sie noch nicht da war.

Trinken wir also unsere Amiga-Cocktails, während wir in Amiga-Shirts und Amiga-Jacken dasitzen, holen unsere Amiga-Notizbücher aus der Amiga-Tasche und schreiben mit unseren Amiga-Stiften das Zusatzwort zum Amiga: »Faszinierend«…

(Manfred Kohlen/ev)

Liebe auf den zweiten Blick

Kein 68000er-Computer ist so umstritten wie der QL. Es bedarf schon einiger guter Gründe, dieses Sinclair-Produkt zu kaufen.

Zum besseren Verständnis des QL ein kurzer Blick auf seine Entstehungsgeschichte und seine Wiege. Sinclair Research wurde 1979 von Clive Sinclair in Cambridge/England gegründet, um elektronische Konsumgüter zu entwickeln und zu vertreiben. Der Erfindergeist von Clive Sinclair brachte der Computerbranche mit dem legendären ZX81 im März 1981 eine Revolution. Nie zuvor gab es so billig einen Computer für jedermann, der Heimcomputer begann seinen Siegeszug quer durch alle Kinderzimmer. Der Preis des ZX81 von anfangs 500 Mark fiel im Laufe der Jahre auf 100 Mark und stellte damit keine Hemmschwelle zur Anschaffung eines Computers mehr dar.

Mit steigender Verbreitung der Heimcomputer stiegen auch die Erwartungen, die an ein solches elektronisches Wunderwerk gestellt wurden. Sinclair konzipierte seinen zweiten großen Erfolg, den Spectrum, von dem in Deutschland über 120000 Stück verkauft wurden. Auf Grund der ZX81-Erfolge sprangen viele Software-Entwickler auf den neuen Siegeszug, und es gibt keinen britischen Heimcomputer, für den auch nur annähernd so viele Programme angeboten werden wie für den Spectrum.

Statt Quantensprung Verzögerung

Dennoch waren die Grenzen des Spectrum schon wegen des 8-Bit-Prozessors abgesteckt. Clive Sinclair, im Juni 1983 von der Queen geadelt und damit zu Sir Clive Sinclair avanciert, stellte im Sommer 1983 einen Computer vor, der zu diesem Zeitpunkt seinen Namen verdiente: den Sinclair QL. QL steht dabei für »Quantum Leap«, Quantensprung. Diesen, nach anfänglichen technischen Schwierigkeiten inzwischen ausgereiften Computer mit neuester 16-Bit-Technologie, gibt es nun auch in einer Version zu kaufen, die ganz auf den deutschen Markt abgestimmt ist. Allerdings war dieser Schritt mehr Evolution als Revolution. Durch eine Verkettung von Pannen dauerte der deutsche Quantensprung volle 18 Monate. Spötter interpretieren deshalb nicht ganz zu Unrecht QL mit »Quite Late«, ziemlich spät.

Was ist nun eigentlich dran am QL? Das Besondere an dem QL ist, daß er mit einem 16-Bit-Prozessor arbeitet. Es handelt sich um einen Motorola-68008-Chip, einen Prozessor mit interner 32-Bit-Struktur und einem 8-Bit-Datenbus. Die zweite Besonderheit ist, daß zwei Laufwerke für Microdrive-Cartridges eingebaut sind. Diese kleinen und billigen Massenspeicher sind zwar kein vollwertiger Diskettenersatz, aber ein gewaltiger Fortschritt gegenüber den normalen Datenkassetten hinsichtlich Datensicherheit und Übertragungsgeschwindigkeit. Damit wird neben einem Datensichtgerät (Fernseher oder Monitor) nur noch ein Drucker benötigt, und die Datenverarbeitungsanlage ist komplett. Der dritte bemerkenswerte Punkt ist ein Softwarepaket, das jedem QL beiliegt. Für den IBM-PC beispielsweise kostete es ein Vielfaches des QL-Preises. Dieses Programmpaket zeigt übrigens deutlich, welche Käufergruppe von Sinclair für den QL angepeilt wurde: die kommerziellen Anwender (Rechtsanwälte, Ärzte, Gewerbetreibende). Bei dem Software-Paket handelt es sich um eine hervorragende Textverarbeitung, ein kaufmännnisches Kalkulationsprogramm, eine Datenverwaltung und ein erstaunliches Grafikprogramm zur Darstellung von Zahlenwerken in optisch einprägsamer Form.

Diese Programme sind für die deutsche QL-Version vom Hersteller, dem britischen Edelsoftwarehaus Psion, komplett übersetzt worden.

Der augenfälligste Unterschied zwischen dem britischen Original und der deutschen Version ist natürlich die Tastaturbelegung mit den Umlauten und den deutschen Sonderzeichen. Sichtbar ist der Unterschied auch auf der Rückseite der »Black Box« im typischen Sinclair-Design. Gemeint sind die Anschlüsse der beiden RS232- und der beiden Joystick-Ports. Bei der Version für Deutschland sind dafür 9-polige Normverbindungen vorhanden, die Briten haben exotische Steckbuchsen.

Der QL aus England – nun völlig auf den deutschen Anwender abgestimmt

Das QL-System spricht deutsch

Erst beim Betrieb fällt auf, daß sogar die Fehlermeldungen des ROM in deutscher Sprache erscheinen. Schraubt man verbotenerweise (Garantie-Problem) das Tastaturgehäuse auf, so sieht man einen weiteren deutlichen Unterschied: Die deutsche QL-Version ist gegen Störausstrahlungen abgeschirmt und damit VDE- und FTZ-geprüft.

Die britische Version kostete anfangs 400 britische Pfund (1600 Mark). Der Preis wurde kürzlich auf die Hälfte redu-

ziert und liegt damit umgerechnet nur noch bei knapp 800 Mark. Die deutsche Version, anfangs auf 1700 Mark beziffert, kostet zur Zeit knapp unter 1000 Mark.

Der QL hat in der Grundausstattung ein 48-KByte-ROM, das durch Steckmodule um 16 KByte erweitert werden kann. Der dafür vorgesehene Steckplatz befindet sich an der Rückseite des Gehäuses. Der Schreib-Lese-Speicher (RAM) ist mit 128 KByte bestückt und kann um 512 KByte erweitert werden. Entsprechende Einschubplatinen kosten zur Zeit rund 800 Mark. Vom RAM-Bereich gehen für den Benutzer 32 KByte dadurch ab, daß dieser Bereich als Bildschirmspeicher reserviert ist. Als Datensichtgeräte können beim QL sowohl Fernsehergeräte als auch monochrome oder Farbmonitore mit Video- oder RGB-Anschluß verwendet werden. Wegen der hohen Auflösung von 512 und 256 Bildpunkten empfiehlt sich aber ein guter monochromer oder ein RGB-Monitor. Die wohl preiswerteste Lösung ist die Anschaffung eines Zenith ZVM 1230, eines 12-Zoll-Monochrom-Monitors. Dieser ist problemlos an den QL anzupassen und stellt auch 80 Zeichen pro Zeile lesbar dar. Ein RGB-Monitor mit gleich guter Darstellung kostet rund 1000 Mark mehr, wobei der Nutzen in einem schlechten Verhältnis zum Preis steht.

80 Zeilen auch in Farbe

Für Drucker und andere Ein- oder Ausgabegeräte stehen zwei RS232-Schnittstellen zur Verfügung eine davon ist als Datenendeinrichtung (DTE) und eine als Datenübertragungseinrichtung (DCE) geschaltet. Für Drucker mit Centronics-Schnittstelle ist ein Interface erforderlich.

Zwei Ports für Joysticks und zwei Buchsen für Netzwerk-Betrieb vervollständigen die Schnittstellen. Das Netzwerk ist zum Datenaustausch mit 100 KBaud Übertragungsrate (das sind über 100 000 Bit pro Sekunde) zwischen mehreren QL oder/und Spectrum gedacht. Des weiteren ist ein Fach für Einschübe vorhanden. Hier finden Speichererweiterungen oder Kontroller für Diskettenlaufwerke Platz.

In dem 48-KByte-ROM sind sowohl das sogenannte Superbasic als auch das QDOS, das eigentliche Betriebssystem und der Zeichensatz untergebracht. Das QDOS verwaltet neben dem ROM und RAM auch die beiden internen Microdrives sowie bis zu sechs extern anschließbare Microdrives, die je gut 100 KByte Speicherplatz bieten. Die externen Drives sind mit denen des Spectrum identisch, die Spectrum-Cartridges jedoch sind für den QL nicht lesbar, da ein anderes Aufzeichnungsformat vorliegt. Ein Cartridge kostet zur Zeit etwa 8 Mark und gilt mittlerweile in bezug auf die Datensicherheit als recht zuverlässig.

Basic mit Struktur

QDOS unterstützt auch das sogenannte Multitasking, also das Abarbeiten mehrerer Programme gleichzeitig. Dieses gelingt durch geschickte Arbeitsteilung. Neben dem genannten 68008-Prozessor, der im 7,5-MHz-Takt arbeitet, ist für die Tastatur und die Schnittstellen sowie die akustische Signalarbeit ein 8049-Prozessor als »Co-Pilot« tätig. Als »Bremser« im Arbeitsablauf wirkt der Bildschirmaufbau. Hier gelang keine optimale Lösung. Auf dem Bildschirm sind maximal 512 mal 256 Bildpunkte mit maximal vier Farben oder 256 mal 256 Bildpunkte mit maximal acht Farben darstellbar. Die geringere Auflösung ist für den Betrieb mit einem Fernsehgerät gedacht, ein Monitor schafft auch die 512 Punkte je Zeile.

Der Name Superbasic ist für das Basic des QL richtig getroffen. Es ist einfach super, und es findet sich kaum Vergleichbares in den Reihen der Mitbewerber. Da ist zuerst die Möglichkeit zu nennen, sogenannte Prozeduren zu definieren. Wie man es vom Pascal her kennt, wird eine einmal definierte Prozedur wie ein Basic-Befehl mit Namen aufgerufen und abgearbeitet. Dadurch werden Subroutinen, die mit GOSUB/RETURN oder GOTO angesprungen werden, faktisch überflüssig, und ein Listing gewinnt an Übersichtlichkeit. Dennoch sind diese Befehle im QL-Basic enthalten, um die Umsetzung von Programmen anderer Basic-Dialekte zu erleichtern.

Erwähnenswert sind auch die Grafikbefehle, die zum Teil an die Programmiersprache Logo erinnern. Werden die Basic-Befehle PENUP, MOVE, TURNTO, FILL, SCALE und PAN um einige Prozeduren (faktisch eigene Basic-Befehle) angereichert, hat man schnell seine Schildkröten-Befehle beisammen und den Grundstein für ein eigenes Grafikprogramm gelegt. Es lassen sich dann auch noch mit der Window-Technik verblüffende Ergebnisse erzielen. Window-Technik bedeutet, daß Sie den Bildschirm in beliebige »Windows«, also Fenster, aufteilen, in denen dann gleichzeitig (Multitasking) verschiedene Programme ablaufen. Die Tabelle der reservierten Basic-Worte (Befehle und Operationen), gibt einen Eindruck von der Vielseitigkeit. Superbasic ist wohl einmalig bei einem Computer dieser Preisklasse.

Die Hauptsache als Zugabe

Wie bereits erwähnt, bekommt jeder QL-Käufer ein Software-Paket mitgeliefert. Beim englischen QL natürlich in englischer Fassung, bei der deutschen QL-Ausgabe komplett in Deutsch. Daß diese Programme in englischer Version unter dem Namen Xchange von Psion für Personal Computer (zum Beispiel IBM-PC) zu einem Preis von umgerechnet rund 1600 Mark angeboten werden, spricht wohl für sich. Dies ist mehr als das Doppelte des derzeitigen Preises für die britische QL-Ausgabe inklusive dieser Software.

Das Programmpaket umfaßt den Bereich der Textverarbeitung, der Kalkulation, der Geschäftsgrafik und der Datenverwaltung. Egal welches Programm Sie laden, Sie finden sich

Name	Sinclair QL
Mikroprozessoren	Motorola 68008 und Intel 8049
Taktfrequenz	7,5 MHz
Speicher	128-KByte-RAM, erweiterbar auf 640-KByte-RAM
Tastatur	QWERTZ, Umlaute, 65 Tasten
Massenspeicher	2mal 100-KByte-Microdrives
Auflösung	512 x 256 Bildpunkte mit 4 Farben
	256 x 256 Bildpunkte mit 8 Farben
Textdarstellung	40 bis 60 Zeichen pro Zeile
Schnittstellen	2 serielle RS232C-Schnittstellen
	Monitor- und TV-Ausgang
	2 Joystick-Ports
	ROM-Karten-Anschluß
	Steckplatz für Einschübe
	2 Anschlüsse, um mehrere QL zu verbinden
Software.	»Quill« (Textverarbeitung)
(im Lieferumfang enthalten)	»Abacus« (Kalkulation).
	»Archive« (Datenverwaltung)
	»Easel» (Geschäftsgrafik)
Preis:	Knapp unter 1000 Mark

Daten und Preis des QL

schnell darin zurecht. Das liegt unter anderem daran, daß das »Design«, also der optische Aufbau, und die Befehlsstruktur fast identisch sind. Hier ist Psion wirklich ein großer Wurf gelungen. Und wenn Sie dann doch einmal ratlos sind, ein Druck auf die Funktionstaste 1 hilft in jeder Programmsituation mit einer umfassenden Anleitung weiter

Das wohl wichtigste Programm aus dem Software-Paket ist Quill, eine Textverarbeitung. Sie zeichnet sich durch eine hervorragende Arbeitsgeschwindigkeit beim Umformatieren, beim Suchen und Verschieben von Textblöcken sowie beim Einfügen von einzelnen Zeichen, Worten oder gar Absätzen aus. Neben dem Druck in normaler Schrift stehen Fettdruck, Unterstreichen und Hoch- oder Tiefstellen zur Auswahl. Auch Kopf- und Fußzeile werden auf Wunsch gedruckt und mit Zahlen versehen. Die Anpassung Ihres Druckers wird kaum Probleme aufwerfen; bereits im mitgelieferten Installationsprogramm sind elf verschiedene Druckertreiber vorhanden. Einer dieser Treiber spricht sicherlich auch Ihren Drucker an. Sollten Sie einen Exoten mit absonderlichen Steuerbefehlen als Drucker verwenden, dann können Sie sich im Dialog mit dem Installationsprogramm einen speziellen Druckertreiber zusammenstellen.

QL komplett mit Software

Das für Demonstrationen des QL am besten geeignete Programm ist Easel, das Geschäftsgrafik-Programm. Hiermit sind Zahlen, die in Tabellenform wenig beeindrucken, optisch sehr einprägsam darstellbar. Easel ist auch das einzige der vier Grundprogramme, das die Anschaffung eines Farbmonitors rechtfertigt. Die darzustellenden Zahlenwerke können sowohl direkt in das Programm eingegeben als auch per Befehl aus einer Abacus-Datei übernommen werden.

Abacus ist der Name des Kalkulationsprogramms für speziell kaufmännische Aufgaben. Die »Tabelle« kann aus bis zu 255 Zeilen mit je 64 Spalten bestehen. Jede Zeile und Spalte bekommt einen Namen zugewiesen. Damit ist jede der maximal 16320 Zellen ansprech- und füllbar. Der Befehlsschatz von Abacus ist recht umfangreich und erlaubt sogar Balkendiagramme. Wer auf einer besseren grafischen Darstellung besteht, der übergibt das Zahlenwerk an Easel zur Auswertung und zum Ausdruck der Bildschirmgrafik über einen Matrixdrucker.

Als letztes aus dem Psion-Quartett ist Archive zu nennen. Archive ist eine Datenbank oder, besser gesagt, ein Programm zum Entwurf von individuellen Datenbanken. Archive stellt eine eigene, leicht zu lernende Programmiersprache zur Verfügung. Die Sprache ist so leistungsfähig, daß sich sogar Lagerverwaltungsprogramme mit Umsatzerrechnung und Statistik »zusammenstellen« lassen. Im Gegensatz zum Basic sind hier erfreulicherweise alle Programmierbefehle in deutsch.

Kein Computer ohne Umfeld

Das Umfeld des QL teilt sich in Software und Hardware. Auf der Softwareseite ist zur geistigen Erbauung das QL-Chess, ein sehr gutes und grafikstarkes Schachprogramm, erwähnenswert. Dieses Psion-Programm sollte dem QL-Besitzer nicht fehlen. Auch imponiert die Vielzahl der angebotenen Programmiersprachen für den QL. Diese und weitere Softwareangebote entnehmen Sie bitte der Marktubersicht an anderer Stelle. Wichtigste Hardware um den QL

COS
SIN
TAN

ATAN
ACOT

ACOS
ASIN

COT
EXP
LN
LOG10

INT
ABS

RAD
DEG

PI

RAND
RANDOMISE

Liste der mathematischen Funktionen

ABS	DEFine PROCedure	LEN	RANDOMISE
ACOS. ASIN	END DEFine	LET	RND
ACOT, ATAN	DEG	LIST	RECOL
ADATE	DELETE	LOAD	REMark
ARC, ARC_R	DIM	LOCal	RENUM
AT	DIMN	LN, LOG 10	REPeat,
AUTO	DIR	LRUN	END REPeat
BAUD	DIV	MERGE	RESPR
BEEP	DLINE	MOD	RETurn
BEEPING	EDIT	MODE	RETRY
BLOCK	ELLIPSE,	MOVE	RUN
BORDER	ELLIPSE_R	MRUN	SAVE
CALL	EOF	NET	SIN
CHR$	EXEC, EXEC_W	NEW	SCALE
CIRCLE	EXIT	NEXT	SCROLL
CIRCLE_R	EXP	ON GO TO	SDATE
CLEAR	FILL	ON GO SUB	SElect
CLOSE	FILL$	OPEN, OPEN_IN	END SElect
CLS	FLASH	OPEN_NEW	SEXEC
CODE	FOR	OVER	SQRT
CONTINUE	END FOR	PAN	STOP
RETRY	FORMAT	PAPER	STRIP
COPY, COPY_N	GO SUB	PAUSE	TAN
COS	GO TO	PEEK, PEEK_W	TO
COT	IF, THEN, ELSE	PEEK_L	TRA
CSIZE	END IF	PENUP	TURN
CURSOR	INK	PENDOWN	TURN TO
DATA, READ,	INKEY$	PI	UNDER
RESTORE	INPUT	POINT, POINT_R	VER$
DATE$, DATE	INSTR	POKE, POKE_W	WIDTH
DAY$	INT	POKE_L	WINDOW
DEFine FuNction,	KEYROW	PRINT	
END DEFine	LBYTES	RAD	

Liste der reservierten Basic-Worte

sind Speichererweiterungen und für Microdrive-Feinde Diskettenlaufwerke mit QL-Interfaces. Sinclair bietet hier unter seinem Logo im QL-Design ein 3,5-Zoll-Diskettenlaufwerk um etwa 1000 Mark an, das jedoch nicht für den deutschen Markt bestimmt ist. Hier gibt es offenbar Zulassungsprobleme wegen der Hochfrequenzstörungen.

Speichererweiterungen bis auf 640-KByte-RAM und sogar eine Festplattenstation mit 7,5-MByte-Speicherkapazität von Quest für 4000 Mark runden das Angebot in England nach oben hin ab.

Als Drucker wird von Sinclair im QL-Style ein Matrixdrucker um die 1000 Mark angeboten. Er ist bis auf das ROM baugleich zum Seikosha SP-1000-AS. Aber es eignet sich auch jeder andere Drucker, vorzugsweise mit einer RS232-Schnittstelle (V24). Für Drucker mit Centronics-Schnittstelle ist ein Interface (Schnittstellenwandler) für rund 150 Mark notwendig.

Zum Thema Monitor haben wir uns schon am Anfang dieses Beitrages geäußert und empfehlen dem ernsthaften Anwender einen guten monochromen Monitor. Er sollte aber auf jeden Fall die Möglichkeit der Bildgrößeneinstellung von außen haben. Im Monitor-Modus weicht der Bildaufbau des QL vom Standard dahingehend ab, daß das Bild größer ist. Kann das Bildfenster nicht justiert werden, dann sehen Sie jeweils nur einen Ausschnitt (an allen Seiten fehlt ein Stück). Der von uns getestete Zenith ZVM 1230 ist mit wenigen Handgriffen (Einstell-Räder) an den QL angepaßt. Als Farbmonitor kommt nur ein RGB-Monitor in Frage, der speziell auf den QL abgestimmt ist. Wer hier Geld sparen will, der spart am falschen Ende. Textverarbeitung in Farbe mit 80 Zeichen pro Zeile ist, wenn überhaupt, nur mit hochauflösenden RGB-Monitoren vertretbar.

Preiswert, nicht nur zum Lernen

Der QL in der britischen Version ist für den interessant, der sich in die Programmierung der 68000er-Prozessoren einarbeiten und dafür möglichst wenig Geld ausgeben will. Es ist der mit Abstand billigste 16-Bit-Computer. Der deutsche QL ist durchaus in der Lage, die Erwartungen an eine Datenverarbeitungsanlage zu erfüllen, die in der Heimanwendung an einen Personal Computer gestellt werden. Ob er von kommerziellen Computer-Anwendern akzeptiert wird, bleibt abzuwarten. Ein zu etablierendes dichtes Servicenetz könnte hier den Ausschlag geben.

(Manfred Kotting/hb)

Stride 440 mit Grafikdemonstration - die Kugeln rotieren gegenläufig

Die Cray des kleinen Mannes: Stride

Während der Schwerpunkt dieser Ausgabe bei Computern liegt, die sich vor allem durch eine neuartige Benutzeroberfläche auszeichnen, ist das Spektrum von Computern mit dem Prozessor 68000 sehr breit: Es reicht vom gehobenen Heimcomputer QL von Sinclair über die Low-Cost-Benutzercomputer Atari 520 ST, Mac und Amiga bis hin zu technisch-wissenschaftlichen Supermicros wie der Serie 400 von Stride.

Rod Coleman, »President« von Stride, sagte einmal: »Wir bauen Computer für Leute, die schnelle Computer mögen.«

Kein Zweifel, der Supermicro »Stride« ist so etwas wie eine Cray unter den Micros: 12 MHz CPU (68000), Minimalausstattung von 1 MByte RAM, Ausbaufähigkeit bis 12 MByte und über 400 MByte Massenspeicher (!) in Form von Festplattenlaufwerken, die Fähigkeit, mehreren Benutzern sogar mehrere verschiedene Betriebssysteme gleichzeitig zur Verfügung zu stellen, ein lokales Netz (Omninet) mit einem netzwerkfähigen Betriebssystem (Liaison, eine Weiterentwicklung des UCSD-P-Systems), mehr als ein Dutzend verschiedene Betriebssysteme von CP/M-80 bis Unix V.... Dabei kostet sie nur etwa so viel wie ein Kleinwagen in den kleinen, so viel wie ein Sportwagen in den mittleren Ausbaustufen, in den großen vielleicht soviel wie ein – nun, bei jenem Automobil spricht man nicht über Preise, aber für eine große Stride kann man schon einiges ausgeben.

Alle Versionen verwenden ein und dieselbe CPU-Karte, denselben Bus (VME), dieselbe Software (wobei Unix nur mit entsprechend viel Massenspeicher gefahren werden kann), sie unterscheiden sich nur in der Anzahl verwendeter Erweiterungskarten.

Verglichen mit einem Atari oder Amiga sind Stride-Computer teuer, und sie werden nie Massencomputer sein. Doch wurde viel Software, die auf die-

sen populären Computern verwendet wird, auf Stride-Computern entwickelt. So zum Beispiel der auf dem Atari verfügbare CP/M-Emulator.

So eine Maschine kann sich auch eine kleine Softwarefirma durchaus leisten. Ein Atari dürfte drei- bis sechsmal schneller als ein IBM-PC sein, eine Stride ist aber rein rechnerisch noch 1½mal schneller als ein Atari, effektiv sicher mehr, denn sie hat zum Beispiel wesentlich schnellere Laufwerke. Eine Stride kann zwölfmal mehr Arbeits-Speicher als (gegenwärtig) ein Atari 520 ST+ benutzen.

Das Konzept einer Stride unterscheidet sich sehr von dem der anderen hier vorgestellten Computer. Eine Stride ist eine flache »Blechschachtel« mit einem oder mehreren Laufwerken und einer LED auf der Vorderseite. Auf der Rückseite findet man mindestens vier, maximal 22 serielle Schnittstellen (bis 38400 Baud; Atari oder IBM: max. 9600 Baud) sowie den Stecker für das Netzwerk und eine Parallelschnittstelle (Centronics), einen Netzschalter und einen Reset-Knopf. Um den Computer zu benutzen, muß man ein (oder mehrere) Terminals anschließen. Dagegen verwenden Mac, Amiga und Atari wie konventionelle Personal Computer integrierte Bildschirme, deren Speicher Teil des Arbeitsspeichers des Prozessors ist, was vor allem für die typische Grafikoberfläche dieser Computer von Bedeutung ist. Das ist (für einen Arbeitsplatz pro Maschine) auch mit einer Stride möglich. Die Grafik, die im Bild zu sehen ist, zeigt einen durchlöcherten, rotierenden Ball, in dem ein kleinerer Ball schwebt, der in umgekehrter Richtung rotiert. Diese Demonstration ermöglicht eine Zusatzkarte, die ebenfalls auf den Arbeitsspeicher des Prozessors zugreift und die mit einem besonderen Terminal (Wyse 50) zusammenarbeitet, wobei pro Sekunde aus dem Hauptspeicher bis zu 58 Bildphasen ausgegeben werden können. Die Auflösung beträgt 784 x 325 Punkte. Eine andere Lösung verwendet ein Grafiksubsystem (zum Beispiel »GES Term1« von Graf Elektronik) oder ein Grafikterminal (zum Beispiel DEC VT 240 oder Tektronix), damit ist dann auch Farbgrafik möglich. Die Auflösung richtet sich nach dem Grafikterminal, allerdings ist eine solche Grafik nicht so schnell wie eine, die auf einem Video-RAM im Arbeitsspeicher des Prozessors beruht.

Aber die Bedeutung einer Stride liegt eigentlich woanders – unter der unscheinbar grauen Blechhaut schlummert Spitzentechnologie. Extrem schnelle Laufwerke und Harddisks, ein schier unbegrenzter Arbeitsspeicher, Unix, alle erdenklichen Programmier-

Eigenschaft	Stride 420	Stride 440	Stride 460
Standard Prozessor	Motorola 68000 mit 12 MHz, 3 Millionen Operationen pro Sekunde – ohne Wartezyklus für den Speicher, 32-Bit-Register – 14 Adressierungsarten		
Bus	VMEbus-kompatibel, echte vektorisierte Interrupts, alle Ein-Ausgaben wahlweise mit Polling, 16-Bit-Daten, 24-Bit-Adreßbus, adressiert 16 Megabyte		
Zusätzliche VME-Steckplätze	1 Zusatzkarte	1 Zusatzkarte	2 Zusatzkarten + 2 Europakarten (= 4 Plätze)
Speicher	256 KByte dynamische RAMS (120 ns) mit Paritätsbit ohne Wartezyklen, 4 KByte batteriegepuffertes CMOS-RAM, 16 KByte Booter/Debugger im ROM		
Möglicher Hauptspeicher-Ausbau	1 oder 2 MByte	1, 2, 4 oder 8 MByte	1 bis 8 oder 12 MByte
Floppy Disk	Doppelseitig, doppelte Schreibdichte, 5¼ Zoll, arbeiten ohne Sektor- oder Track-Versatz, daher extrem schneller Zugriff, 640 KByte pro Laufwerk. Ein Laufwerk Standard, wahlweise zweites Laufwerk		
Winchester Kapazität	–	10, 20, 33 oder 67 MByte	33, 67, 112, 224 oder 448 Megabyte
Wahlweise Bandeinheit Streaming Tape Driver	–	45 oder 60 Megabyte pro Kassette, überträgt ein Megabyte in 11 sec, ANSII-Standard X3.55, Qic-02-Format.	
Grafik	Monochrom, 784 x 325 Punkte schwarzweiß, bis zu 58 Bildwechsel/s (optional), oder Anschluß beliebiger Grafikterminals		
Nicky (NOD)	optoelektronisches System, mit dem per Kopfbewegung der Cursor ähnlich wie mit einer Maus gesteuert werden kann (optional)		
Serielle Schnittstellen	4	10 oder 16	10, 16 oder 22
	(mit 300 Baud bis 38,4 Kilobaud)		
Parallel-Schnittstelle	1 Centronics-kompatible Druckerschnittstelle, Zweirichtungsbetrieb mit Pufferung, 8 Daten- und 7 Steuerbits.		
Lokales Netzwerk	Alle Systeme optional mit Omninet-Controller ausrüstbar, erfordert nur Zweidraht-Leitung, 1 Mbps Übertragungsrate, maximal 64 Computer, maximal 500 m Distanz.		
Echtzeituhr	Batteriegepuffert, 0 01% Genauigkeit, Sekunden, Minuten, Stunden, Tage, Monate, Stunden, Jahre		
Wahlweise Gleitkomma-Hardware	Verwendet National 16081 und kundenspezifische PALs – alle Operationen auf 32- oder 64-Bitzahlen, entspricht IEEE-754-Standard Führt 7500 bis 48000 64-Bit-Multiplikationen pro Sekunde aus, typisch 17000 (optional)		
Wahlweise Speicherverwaltungs-Einheit (MMU)	Adressiert 2-Megabyte-Task in 4KByte großen Seiten, ohne Wartezyklen für Speicher auf der CPU-Karte, für Unix System V		
Schaltnetzteil	65 W	140 W	1x oder 2x 140 W
Standard Software	Liaison Operating System, P-System IV 21 mit Netzwerk-Software, Bildschirmeditor, Filemanager, Dienstprogrammen, Disk- und Drucker-Server etc. Single- oder Multiuser-konfigurierbar		
Standard Mehrbenutzer BIOS	Bis zu 15 Benutzer (abhängig von der Ausstattung mit Hauptspeicher und seriellen Schnittstellen), führt verschiedene Betriebssysteme parallel aus, echte parallele Prozesse, Datentausch zwischen verschiedenen Benutzern über Liaison-Netzwerk.		
Weitere Betriebssysteme	CP/M-68K, RM-COS, UNIX V, Idris, HyperFORTH, PDOS, BOS, Mirage, MOSYS (Modula Operating System), TRIPOS, CP/M-80-Emulator		
Weitere Sprachen	68000 Assembler, APL, Basic, C, CBasic, COBOL, Forth, Fortran, Lisp, Modula-2 (verschiedene), Pascal (z. B. SVS-Pascal), SVS-Fortran		
Anwendungsprogramme	Word/7 Textprozessor, Datafax, Aladin plus Datenbank, Kermit, Mince-Editor (CP/M-68K), Emacs-Editor Unix, Grafik-Editor (UCSD)		

Technische Daten Stride Serie 400

sprachen, Floating-Point-Prozessor, diese und andere Merkmale machen sie zu einem technisch-wissenschaftlichen Entwicklungsrechner. Der Computer wird an Universitäten ebenso verwendet wie in Ingenieurbüros oder von Softwareentwicklern

Die Stride-Computer sind durch die Verwendung von allgemein verbreiteten Betriebssystemen wie UCSD (Apple II, IBM-PC, VAX), CP/M-68K oder Unix mit vielen anderen Computern, meist sogar mit anderen Prozessoren, kompatibel, soweit es die Software-Entwicklung betrifft. Daß sie kaum über modellspezifische Eigenschaften verfügen wie etwa integrierte Bildschirme mit bestimmter Auflösung oder bestimmten Text-Fonts, sondern Terminals verwenden, keine Spezialschnittstellen, sondern serielle Schnittstellen (RS232C) und Centronics benutzen, unterstreicht noch ihre neutrale Rolle als technisch-wissenschaftliche Entwicklungsrechner. Sie werden aber auch als Prozeß- und Anwendungsrechner verwendet, beispielsweise gibt es ein Projekt, bei dem eine Stride im Rechnerverbund eine Verkehrssteuerungsanlange an einem besonders nebelgefährdeten Autobahnabschnitt steuert. Eine andere typische Anwendung ist eine Baustatik-Software mit grafisch unterstützter Eingabe von Bau-Grundrissen, die mit verschiedenen angeschlossenen Plottern neben Statik-Berechnungen Produktionszeichnungen für die Beton-Platten-Fertigung erzeugt. Die zugehörige Software wurde auf einer Maschine mit mehreren Terminals von einem Programmierer-Team entwickelt. Durch den VMEbus, eine teure, aber standardisierte Hochleistungs-Technologie, können die verschiedenartigsten Zusatzkarten eingesetzt werden.

Standardmäßig wird eine Laufzeitversion (ohne Compiler) des erweiterten UCSD-P-Systems von Softech mit Netzwerk- und Mehrbenutzerfähigkeit mitgeliefert. Dazu gibt es die P-Compiler für Pascal (Module, Prozesse, ISO-Standard), Fortran und Basic. Ferner werden verschiedene Texteditoren, Datenbanksysteme und zahlreiche Betriebssysteme angeboten (siehe Tabelle). Besonders schnellen Maschinen-Code erzeugt zum Beispiel SVS-Pascal (CP/M-68K, Unix), ein Native-Code-Compiler, dessen Syntax und Sprachmerkmale weitgehend UCSD-Pascal entsprechen. Durch den mittlerweile auf dem Atari bekannten CP/MZ80-Emulator kann die Stride auch CP/M-80-Software wie dBASE, Wordstar, Assembler, Debugger, Turbo-Pascal und andere verarbeiten.

Die verschiedenen Ausbaustufen und Ausstattungen mit Betriebssystemen und Sprachcompilern ergeben so viele Kombinationsmöglichkeiten, daß hier nur einige Beispiele für Preise angegeben werden können. Zur Zeit kostet das kleinste Modell 420 (noch mit 512 KByte) mit einem Floppy-Laufwerk 5 Zoll zirka 16800 Mark. Dabei ist ein Laufzeitsystem mit Netzwerkfähigkeit (UCSD-P) inbegriffen. Am oberen Ende steht eine Stride 460 mit 448 MegaByte Harddisk-Kapazität für über 280000 Mark. Die Gleitkomma-Arithmetik kostet 3325 Mark, ein zusätzliches Floppy-Laufwerk knapp 2800 Mark. Pro Megabyte Speichererweiterung werden gut 4200 Mark berechnet. Diese Preise dürften sich angesichts der gegenwärtig zu beobachtenden Marktumwälzung nach unten entwickeln. (le)

Ein Raubtier im schlichten Gewand: Gepard PC

Technik zum »Anfassen«: Gepard-PC

Preislich liegt der Gepard etwa in der Gegend des Macintosh (in vergleichbarer Konfiguration). Vom Konzept her ist er im Gegensatz zum Mac jedoch völlig »offen«: 16 Steckplätze erlauben den flexiblen Ausbau des Systems. Interessenten erfahren software- und hardwaretechnische Details ohne Umschweife. Ein Computer für Einsteiger wie für Profis.

Ursprünglich war der Gepard eine 16-Bit-Ergänzung zum Apple II beziehungsweise zum C 64. Das war sozusagen ein »nackter« Computer – ohne eigene Ein-/Ausgabe-Einheiten. Diese Einsteiger-Konfiguration besteht heute aus CPU-Karte (10 MHz), Busplatine, 512-KByte-RAM und Verbindung zum Heimcomputer sowie umfangreicher Software (Modula-Compiler, 68000-Assembler, Editor und Modulbibliothek). Das Ganze kostet 2598 Mark, wobei eine »unsichtbare« Leistung, nämlich das modulare Konzept für den Ausbau zum Profisystem, mit inbegriffen ist. Bemerkenswert ist hier, daß sich keiner der »großen« Computerhersteller berufen fühlte, solch ein Konzept, das (in Grenzen) im Apple mit so viel Erfolg vorexerziert wurde, weiterzuspinnen.

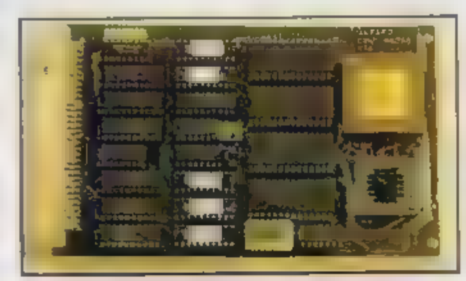
Den eigenständigen Gepard gibt es in zwei Grundversionen: als Text-Paket und als Grafik-Paket. Die 80-Zeichen-Karte der Text-Variante, die für 5798 Mark zu haben ist (ohne Monitor), ist im Grafik-Gepard durch eine GDC-Karte mit einer Auflösung von bis zu 1000 x 700 Punkten ersetzt. Der Aufpreis hierfür beträgt 1000 Mark. Diese Version haben wir uns genauer angesehen.

Optisch macht das »Raubtier«, im Gegensatz zu Konkurrenten wie Atari ST oder Amiga, eher einen bescheidenen Eindruck. »Schlichte Funktionalität« ist wohl der passende Ausdruck dafür. Was jedoch in ihm steckt, muß sich vor den Mitbewerbern keineswegs verstecken: Die 68000-CPU ist standardmäßig mit 10 MHz getaktet, was in dieser Preisklasse sonst nicht zu finden ist. Auf Wunsch gibt es auch eine 12-MHz-Version, die aber nur in Verbindung mit relativ teurem statischen RAM-Speicher Geschwindigkeitsvorteile bringt – dann aber gleich gut 50 Prozent. Weitere Alternativen bietet Gepard in Form eines 68010- und eines 68020-Prozessors. Darauf kommen wir später zurück. Ansonsten umfaßt die Grundausstattung 512 KByte dynamisches RAM, ein 800-KByte-Floppylaufwerk im 3½-Zoll-Format, eine Multifunktionskarte und die schon erwähnte Farb-Grafik-Komponente. An Software kommen zu den Programmen der Einsteigerversion noch das Betriebssystem »Paradise«, eine komfortable Textverarbeitung (Gepstar) und Tabellenkalkulation (Gepcalc). Paradise enthält unter anderem einen Monitor, einen Debugger, einen kleinen Editor und eine ganze Menge Utilities (unter anderem eine RAM-Disk). Der Modula-Compiler ist vollständig in 68000-Assembler-Code geschrieben. Bei dem Textverarbeitungs-Programm »Gepstar« kann man sich mit einem Installations-Modul die Befehle auf beliebige Tasten legen. Wer also Lust hat, arbeitet damit genauso wie mit dem weitverbreiteten Wordstar. Zum Software-Service gehört auch, daß jeweils die neuesten Versionen dieser Programme zum Selbstkostenpreis des Herstellers bezogen werden können. Damit haben sowohl Programmierer als auch Anwender von Standardsoftware gleich beim Kauf eines Gepard ein umfangreiches Sortiment an professionellen Werkzeugen. An weiterer Software stehen bislang das Betriebssystem CP/M-68K (1225 Mark), ein C-Compiler (Preis auf Anfrage), ein Forth-Compiler (900 Mark) sowie »Platool«, ein Layout-Programm für Platinen, zur Verfügung.

Der Floppycontroller basiert auf dem WD 2793-Prozessor und steuert bis zu vier Laufwerke, die sich in beliebiger Mischung aus 3½-Zoll- und 5¼-Zoll-Stationen zusammensetzen. Neben dem bereits vorhandenen Diskettenlaufwerk (Sony) hat noch ein weiteres in dem Gehäuse Platz (940 Mark).

Alternativ dazu wird auch eine 25 MByte große Festplatte (von Sony) zum Einbau angeboten, die inklusive Controller 4499 Mark kostet. Plattenkapazität läßt sich auch extern an den Computer anschließen: Für 5692 Mark gibt es ein Festplattensubsystem (ebenfalls 25 MByte, unformatiert), in dessen Gehäuse wahlweise eine weitere Festplatte (2422 Mark) oder ein Streamerlaufwerk (von Cipher) mit 32 MByte (2458) zur Datensicherung Platz finden.

Die Floppy-Laufwerke arbeiten fast lautlos und übertragen die Daten mit 500 KBit pro Sekunde an den Computer. Die hohe Geschwindigkeit fällt schon beim Laden des Betriebssystems auf: Die etwa 70 KByte sind in »paradiesischen« drei Sekunden im Hauptspeicher. Ein weiterer Geschwindigkeitsvorteil ergibt sich aus der Tatsache, daß das Inhaltsverzeichnis einer Diskette beim ersten Zugriff in den Hauptspeicher geladen wird und dort bleibt, bis die Diskette gewechselt wird. Jeder Wechsel wird sofort und automatisch bemerkt.



Gepards neuestes Glanzstück: Einsteckplatine mit 68020-Prozessor

Bei der Multifunktionskarte fällt vor allem die ungewöhnlich üppige Bestückung auf. Neben dem Anschluß für die Tastatur befinden sich dort eine batteriegepufferte Uhr, zwei Joystick-Anschlüße (dort werden übrigens auch »Mäuse« angeschlossen), zwei 8-Bit-Digital/Analog-Wandler für Musik und Sprachausgabe sowie drei programmierbare Timer für Interrupts. Die Uhr verwendet beispielsweise das Betriebssystem: Im Inhaltsverzeichnis der Disketten stehen bei jeder Datei Datum und Uhrzeit des letzten Update.

Die Tastatur des Gepard hat DIN-Format, LED-Anzeiger, 15 Funktionstasten, einen eigenen Cursor-Block und einen prellfreien Anschlag. Etwas gewöhnungsbedürftig ist lediglich die Anordnung der Cursor-Tasten, mit den Pfeilen für horizontale Bewegung auf der linken, für vertikale Bewegung auf der rechten Seite der Leertaste. Ein kleiner Nachteil ist auch, daß sie keine Füßchen oder dergleichen zum Schrägstellen hat.

Mit der GDC-Grafik-Karte erreicht der Gepard – sofern man von der Verwendung von Farben absieht – eine Auflösung von 1000x700 Punkten. Der Grafikprozessor 7220 von NEC bringt auch bis zu 256 Farben auf den Bildschirm – die Auflösung beträgt dann aber nur noch 200x170 Punkte. Wer mehr möchte, steckt einfach mehrere Grafik-Karten im »Huckepack-Prinzip« zusammen. Bis zu acht solcher Karten – jede mit einem eigenen Grafikprozessor und 128 KByte Video-RAM ausgestattet – lassen sich auf diese Weise »verkuppeln«. Voll ausgebaut verfügt der Gepard also über ein ganzes MByte an Bildschirmspeicher und bringt dann die maximale Auflösung auch mit 256 Farben. Ganz billig wird die Sache nicht, denn für jede Grafik-Karte muß man knapp 1000 Mark bezahlen – insgesamt also runde 8000 Mark allein für die Grafik. Andererseits kosten Grafik-Systeme, die solch eine Auflösung in 256 Farben bringen, üblicherweise ein Vielfaches. Insbesondere wegen seiner Grafikfähigkeit hat der Gepard auch schon in Universitäten, zum Beispiel zur Längenmessung auf Mikroskop-Bildern, und in kleinen Videostudios (als Titelmaschine) Einzug gehalten.

An weiteren Vertriebswegen ist der Verkauf von »OEM-Gepards«, zum Einbau in industrielle Anlagen, sowie die Vermarktung über Softwarehäuser, die den Gepard als Komplettlösung (Hard-/Software) vertreiben, ins Auge gefaßt. Durch die Bildung von Benutzergruppen (eine Liste von Gepard-Anwendern wird schon jetzt jedem PC beigelegt) möchte man den Kreis der engagierten Anwender verstärkt unterstützen.

Für den Gepard gibt es einen ganzen Katalog von Erweiterungen. Will man mit dem Computer Meß-Systeme steuern, steckt man einfach eine IEEE-488-Karte (990 Mark) in einen freien Slot. Pro Karte sind 31 Geräte adressierbar. Für Musikfreunde gibt es ein Midi-Interface (Preis auf Anfrage) mit acht Ausgängen und sechs Eingängen, die alle einzeln ansprechbar sind. Weiter werden ein EPROM-Brenner, ein Analog/Digitalwandler, eine 80-Kanal-Input/Output-Karte und vieles mehr angeboten.

Unerläßlich für den Betrieb eines Computers ist ein »Datensichtgerät« – also ein Monitor. Dieser muß beim Gepard extra dazugekauft werden. Der Vorteil dabei ist, daß man auch hier – wie bei den »Innereien« – frei nach Bedarf beziehungsweise Geldbeutel wählen kann. Für bescheidenere Ansprüche bietet Gepard zwei monochrome Taxan-Modelle an, die man für 399 beziehungsweise 429 Mark bekommt. Farbmonitore gibt es ab 1839 Mark (von Eizo) bis 10445 Mark (von NEC) für Grafik-Profis.

Flexibel ist der Gepard – was wohl jetzt niemanden mehr wundern wird – auch bei den anschließbaren Druckern: Gepard hat hier eine Reihe von Modellen der Data Produkts in den Vertrieb übernommen. Der billigste Matrix-

Drucker (mit 180 Zeichen/s) kostet 2086 Mark. Den Typenrad-Drucker gibt es wahlweise mit oder ohne bidirektionalen Traktor. Der Preis liegt bei 2275 beziehungsweise 2787 Mark.

Auf der Systems stellten die Gepard-Entwickler ihr neuestes »Glanzstück« vor: eine Einsteckplatine mit 68020-Prozessor. Hier handelt es sich um eine echte 32-Bit-CPU, die im Gepard mit 16 MHz getaktet wird. Anwenderprogramme laufen etwa 100 Prozent schneller als mit dem »normalen« 68000er. Weitere 100 Prozent Geschwindigkeits-Steigerung lassen sich erreichen (bei rechenintensiven Programmen), wenn man den freien Sockel der Platine mit dem Mathematik-Prozessor 68881 bestückt (Herstellerangaben). Allerdings fehlt es bislang noch an Software, die die Leistung dieser Prozessoren effektiv nutzt. Die Preise der Chips sind derzeit heftig in Bewegung (nach unten), so daß man die Preise für die Platine bei Gepard, die sofort auf solche Änderungen reagiert, zum jeweiligen Zeitpunkt erfragen muß. Es sieht auf jeden Fall so aus, daß man demnächst einen Personal Computer, der mehr leistet als eine VAX 780, für weniger als 10000 Mark bekommen wird. Mit dem Amiga und dem Atari ST steht dem Gepard harte Konkurrenz ins Haus. Die vergleichbar konfigurierten Systeme sind dort zum Teil erheblich billiger. Bleiben als Vorteile für den Gepard neben einer Reihe technischer Details noch die ungeschlagene Flexibilität und Ausbaufähigkeit sowie der Vorsprung bei der Entwicklung eines 68020-Systems. Damit darf man beim Gepard – wenn auch sicher nicht in den Stückzahlen der populären 68000-Computer – gute Marktchancen erwarten. (sm)

Für jeden etwas

Wir sehen, daß die vorgestellten Computer durchaus sehr unterschiedlichen Anwendergruppen gerecht werden. Der Macintosh ist wohl der seriöseste unter ihnen. Mit dem unverkennbaren Design gehört dieser Computer zum mittlerweile arrivierten »Seniorenfreak«. Er verleiht seinen Besitzern immer noch leicht unterkühltes Renommee, und sei's, weil er teuer ist.

Für Freaks im Sinne von Bytekrabblern und übernächtigten Programmierfixern ist er ungeeignet. Da kommt ein Stride oder ein Gepard schon eher in Frage. Zwischen diesen beiden entscheidet eigentlich nur das Portemonnaie. Wer sich mit ihnen abgibt, darf sich in aller Bescheidenheit zu den echten Insidern zählen – und wird gerade deshalb wenig Wert darauf legen, wie die Umwelt ihn einschätzt. Außerdem ist gerade der Stride die Einstiegsmaschine für die Profis unter den Insidern, jene, die nicht nur davon träumen, eine eigene Softwareschmiede aufzuziehen, sondern diesen Traum auch konkret angehen. Eben ein ernsthaftes Entwicklungssystem.

Der QL scheint etwas für Optimisten zu sein, die auch die innere Unsicherheit um künftige Serviceleistungen und Softwareversorgung ertragen. Dennoch, der QL ist sein Geld wert. Für gewiefte Hardwarebastler, die gleichzeitig viel selbst programmieren, Selbstversorger also, kann dies durchaus der Computer ihrer Wahl sein.

Der Amiga ist ein Phänomen. Er scheint nur glühende Verehrer oder fanatische Gegner zu besitzen. Darin hat er das Erbe seines kleinen Bruders, des C 64 angetreten. Nüchtern betrachtet ist er eine sehr leistungsfähige Maschine, die im Prinzip preiswert und damit Meister aller Klassen wäre – wenn es da nicht den Atari ST gäbe.

Während die Schöpfer des Amiga noch auf den wahren (klingenden) Wert modernster Technologie pochen und den Amiga damit für viele potentielle Anwender wie die berühmten Trauben in der Fabel vom Fuchs zu hoch an den Ast hängen, fällt einem zum ST sofort der Vergleich mit dem Volkswagen ein.

Amiga und ST sind beide die ersten Universalcomputer und für Freaks wie für Anwender fertiger Programme gleichermaßen geeignet. Ihrer bescheidenen Mechanik und der noch fehlenden umfassenden Softwarebibliothek wegen, eignen sie sich derzeit nur für intensive Bürobenutzung noch nicht. Als Anwender kommen daher in erster Linie Privatleute oder Institute in Frage, bei denen in der Regel rechenintensive, aber keine tastenintensiven Arbeiten anstehen

Am Markt findet übrigens insgesamt gesehen ein bisher einmaliges Ereignis statt, das viele nur deshalb nicht wahrnehmen, weil ein Computer nach außen immer nur so gut zu sein scheint, wie die Software ist, die darauf läuft. Eine mehrfach leistungsfähigere Technologie wird seit einem Jahr zu einem geringeren Preis feilgeboten als die derzeit immer noch als Standard akzeptierte. Während man heute nach wie vor 8-Bit-Bürocomputer mit 64 KByte und Pseudo-16-Bit-Computer mit dem 8088-Prozessor plus 256 KByte für über 5000 Mark zu kaufen bekommt, erhält man für 3000 Mark einen Computer mit dem 4fachen Speicher, einem mehrfach so schnellen und leistungsfähigen Prozessor und einer exzellenten Grafik.

Dabei ist das, was wir jetzt auf diesen Maschinen zu sehen bekommen, nur gebremster Schaum. Die Computer-Veteranen unter uns haben schon mehrmals die Erfahrung machen müssen, daß am Anfang einer neuen Computergeneration immer nur ein Bruchteil der wahren Fähigkeiten zu erkennen ist. Erst nach rund zwei Jahren kommen langsam die wirklich guten Programme auf den Markt, die einen Computer ganz ausreizen. Zuletzt bewies sich diese Entwicklung am Commodore 64. Wer erinnert sich noch an die ersten Programme? Grobe Kästchengrafik und langweilige Animationen füllten die Monitore. Und heute? Da gibt es Grafikprogramme mit allen Schikanen und Software für Gott und die Welt. Wir werden dieses Aha-Erlebnis bei den 68-Tausendsassas wieder erleben, sogar im Quadrat. Wetten daß?.... (lg)

Prozessor 68000, CPU für neue Computerkonzepte

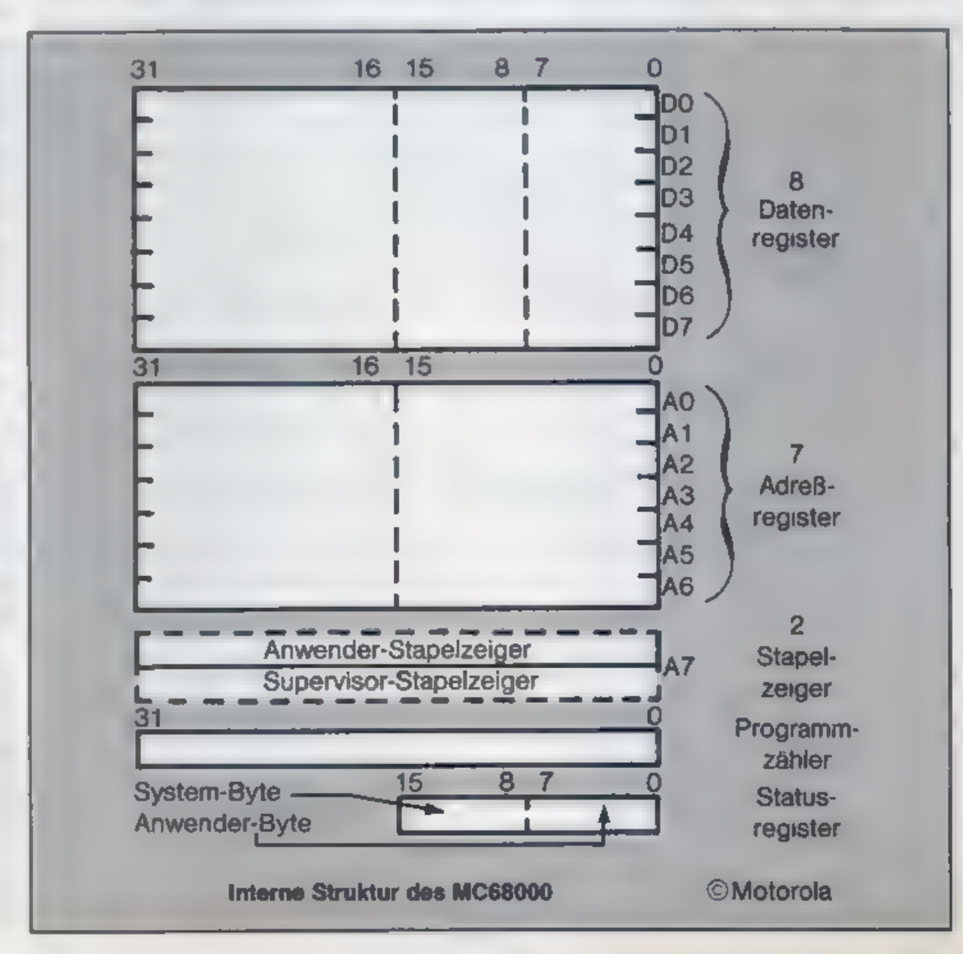
Ein Prozessor, der bislang hauptsächlich in technischen Anwendungen eingesetzt wurde, findet in vielen Computern ein neues Aufgabengebiet.

So ganz neu ist der 68000er von Motorola nicht. Bereits im Jahr 1979 lagen die ersten Prototypen dieses Prozessors vor, der den Entwicklern von Computern erlaubte, die bis dahin bestehenden Grenzen und Einschränkungen ohne große Probleme zu überwinden. Für viele Einsatzgebiete reicht ein 8-Bit-Prozessor aus, was die große Anzahl von CP/M-Computern belegt. Schon mit einem Speicher von nur 64 KByte RAM lassen sich so aufwendige Software-Produkte wie »Wordstar« und »Supercalc« einsetzen. Da die Computer aber in immer breitere Bevölkerungsschichten vordringen, mußten sie immer einfacher zu bedienen sein. Dies führte zur Entwicklung unterschiedlichster Benutzeroberflächen. Allen gemein ist, daß sie verstärkt grafische Elemente einsetzen. Diese benutzerfreundlichen Schnittstellen setzen nur eins voraus: einen Speicherbereich, der deutlich die 64-KByte-Grenze überschreitet. Für eine 8-Bit-CPU müssen schon aufwendige Hardware-Strukturen bereitgestellt werden, damit sie durch »Bankswitching« 128 KByte adressieren kann. Der Weg zu speicherintensiven Benutzeroberflächen führte also zwangsläufig zu einem 16-Bit-Prozessor wie dem 68000, der einen Speicher von 16 MByte direkt adressieren kann. Durch Hinzunahme der Anschlüsse für die Funktionscodes, ist der Bereich auf 64 MByte erweiterbar. Da der 68000er intern mit 32-Bit-Registern arbeitet, wären sogar Speicher von vier GByte direkt adressierbar. Die Entwickler der CPU führten aber »nur« 24 Bit des Adreßregisters heraus, womit der Speicherbereich auf 16 MByte beschränkt bleibt. »Nur« deshalb in Anführungszeichen, weil 16-Bit-Prozessoren bislang mit einem gemultiplexten Bus arbeiteten. Die weitverbreitete CPU 8086 zum Beispiel, die in vielen MS-DOS-Computern ihren Dienst tut, kann nur 64 KByte Speicher direkt adressieren. Erst durch Segmentregister wird die CPU fähig, ein MByte große Speicher zu verwalten.

Durch die 24 Adreßleitungen läßt sich also ein genügend großer Speicher adressieren, der auch für grafische Benutzeroberflächen wie GEM (Atari) oder Intuition (Amiga) ausreicht. Das alleine genügt aber für die Verwaltung aufwendiger Software nicht. Ein weiteres, ausschlaggebendes Argument für die Leistungsfähigkeit eines Mikroprozessors ist die Anordnung der Register und die Verwaltung von Interrupts.

Bei den Registern bietet der 68000er acht Daten- und sieben Adreß-Register, sowie einen Stackpointer und einen Programmzähler, die alle 32 Bit breit sind. Neben diesen 17 Registern hat die CPU noch einen 32 Bit breiten Programmzähler und ein 16 Bit breites Statusregister integriert. Eigentlich hat der 68000er zwei Stackpointer aufzuweisen. Davon ist aber immer nur einer aktiv, was durch das Statusregister gesteuert wird. Im Statusregister wird zwischen acht Anwender- und acht System-Bits unterschieden. Das 13. Bit in diesem Register ist verantwortlich für die Umschaltung der zwei grundsätzlichen, vom 68000er unterstützten, Betriebsarten. Es wird als »S«-Bit bezeichnet, was soviel wie Supervisor-Modus bedeutet. Ist dieses Bit gesetzt, wird das Register A7, der Stackpointer, im Supervisormodus genutzt. Ist das Bit nicht gesetzt, wird der Stackpointer im normalen Anwendermodus betrieben. Zur genauen Beschreibung der einzelnen Register und der unterschiedlichen Modi beginnen wir mit den Datenregistern.

Der Datenbus ist ein 16-Bit-Datenbus. Auf ihn entfallen die Anschlüsse »D0« bis »D15«, mit denen bis zu 16 Bits parallel übertragen werden können. Aber auch 8-Bit-Daten, ein Datenwort also, können verarbeitet werden. Typischerweise ist der Datenbus des 68000er ein bidirektionaler Bus. Daten werden sowohl in den Prozessor geladen, als auch vom Prozessor in das System. Zur Unterscheidung, ob ein 16- oder ein 8-Bit-Wort übertragen wird, dienen die Leitungen »UDS« und »LDS« (Pin 7 und 8). Mit den bei negativer Ansteuerung aktiven Signalen »Upper Data Strobe«, beziehungsweise »Lower Data Strobe«, erfolgt eine



Interne Struktur des MC68000

genaue Zuordnung, welches Byte des 16-Bit breiten Datenbusses eingesetzt werden soll. Dabei wurde festgelegt, daß als unteres Byte die acht Bit der Datenleitungen »D0« bis »D7«, und als oberes Byte die Daten »D8« bis »D15« anzusehen sind. Intern sind die acht Datenregister so organisiert, daß jedes Wort (zwei Byte) auch zwei Adressen belegt. Dabei steht das höherwertige Byte eines Wortes an den niedrigeren Adressen. Die Datenregister werden immer von den niederwertigen Bits an aufgefüllt. Enthält ein Datenregister zum Beispiel ein 16-Bit-Wort, sind davon nur die Bits »D0« bis »D15« betroffen. Die Bits »D16« bis »D31« können mit einem 16-Bit-Wort nicht angesprochen werden.

Der 68000er ist mit sieben Adreßregistern ausgestattet, die alle 32-Bit breit sind. Damit wäre rein theoretisch ein Speicher von vier GByte direkt adressierbar. Aus dem Gehäuse werden aber nur 23 Adreßleitungen herausgeführt (A1 bis A23), die 24. Adreßleitung, die den Prozessor befähigt, 16 MByte direkt anzusprechen, wird durch die Signale »UDS, LDS« erzeugt.

Ein weiteres, wichtiges Signal ist »AS« (Adress Strobe, Pin 6). Diese Leitung zeigt den externen Bausteinen an, daß eine Adresse vollständig auf dem Adreßbus ansteht.

Ein besonderes Register ist das Adreßregister »A7«, auch als Stackpointer oder Stapelzeiger bezeichnet. Es wird bei normalem Betrieb des Prozessors bei Unterprogrammaufrufen als Speicher der Rückkehradresse eingesetzt. Ist aber das »S«-Bit des Statusregisters gesetzt, wodurch der Supervisor-Modus angeschaltet wird, fungiert das Register als »A7'«-Stapelregister für den Supervisor. Durch entsprechende Programmierung sind die Adreßregister »A0« bis »A6« auch als vom Benutzer definierte Stackpointer zu verwenden.

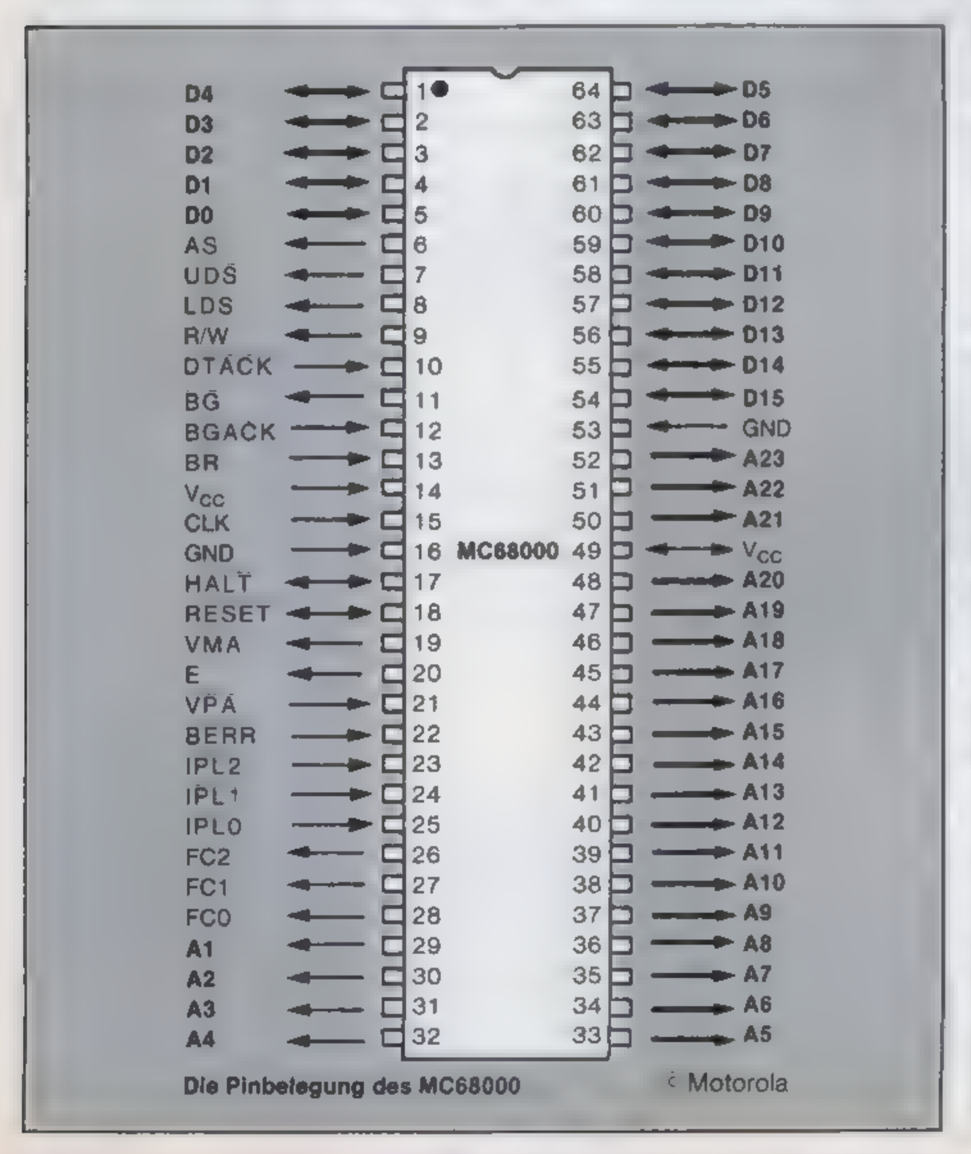
Daten-, Adreß- und Stack-Register sind 32 Bit breit. Durch die hohe Anzahl dieser Register erreicht der Prozessor ein ungewöhnlich gutes Laufzeitverhalten. Er kann dadurch immer mehrere Variable in den Registern bereithalten und verändern, ohne sie erst aus den externen Speichern holen zu müssen. Der Einsatz von Adreßregistern als Datenspeicher, wiewohl auch der Datenregister als Indexregister, ist erlaubt.

Ein weiteres 32-Bit-Register ist der Programmzähler »PC«, von dem die unteren 24 Bit als Adressen eingesetzt werden. In dem Moment, in dem ein Opcode geholt wird, werden die 24 Bit auf den Adreßbus gelegt.

Als letztes der internen Register des 68000er ist das Statusregister zu beschreiben. Es ist zwar nur 16 Bit breit, dafür aber umso wichtiger für das gesamte Konzept des Prozessors. Die unteren acht Bit des Statusregisters bezeichnet man als Anwender-Byte oder Condition Control Register (CCR). Hier finden sich die bei allen Prozessoren notwendigen Flags, die sie in bestimmten Situationen setzen. Etwa das Carry-Flag »C«, das anzeigt, daß bei einer Operation ein Übertrag entstanden ist (Bit 0 des CCR). Die nachfolgenden Bits enthalten das Überlauf-, Zero-, Negativ- und Extend-Flag. Die einzelnen Flags sind übliche »Zustandsanzeiger«, weshalb hier eine nähere Beschreibung entfällt. Eine Ausnahme bildet allerdings das Bit 5 des CCR, das Extend-Flag. Es wird zum Beispiel verwendet, um den Inhalt des Carry-Flags über mehrere Operationen bereitzuhalten. Dieses Bit ist aber erst bei der eingehenden Programmierung mit dem 68000er von Bedeutung.

Aus den aufgeführten fünf Flags setzt sich das Anwenderbyte des Statusregisters zusammen. Auch vom Systembyte werden nur fünf Bit genutzt. Drei entfallen auf die Interrupt-Maske, eins auf den Supervisormode und eines wird als Trace-Bit bezeichnet. Für die Interrupt-Maske sind die Bits 8, 9 und 19 zuständig, die mit den Anschlüssen IPL0, IPL1 und IPL2 korrespondieren. Dadurch lassen sich sieben Interrupt-Level erreichen. Führen alle Interrupt-Leitungen ein High-Signal, liegt kein Interrupt vor. Um einen Interrupt eindeutig zu definieren, wird das einkommende Signal zwei Taktzyklen lang unterdrückt. Danach quittiert der 68000er den Interrupt und legt den Wert der Prioritätsebene auf den Adreßbus (Bits A1 bis A3) und diverse Steuerleitungen. Die Prioritätsebenen 1 bis 6 sind maskierbar, wodurch Interrupts mit niedrigerer Priorität verhindert werden. Eine genaue Beschreibung der Interrupt-Bedienung des 68000er führt hier zu weit. Die Interrupt-Struktur ist sehr flexibel und stellt dem Anwender 199 Interruptvektoren bereit

Bit 13 des Statusregisters ist das »S«-Bit, das den Supervisor-Mode, der auch als Überwachungs- oder System-Mode bezeichnet wird, anzeigt. Ist dieses Bit gesetzt, arbeitet der Prozessor im Supervisormode, der keinen Ein-



Die Pinbelegung des MC68000 © Motorola

schränkungen unterliegt. Dem gegenüber steht der Anwender-Mode (Bit 13 nicht gesetzt). Hier sind zum Beispiel weder alle Befehle zulässig, noch alle Bereiche zugänglich. Der Betriebszustand des Prozessors wird mit den Leitungen FC0 bis FC2, den Function Codes, angezeigt. Je nach dem anliegenden Signal erkennt man, ob sich der Prozessor im Supervisor- oder im Anwender-Mode befindet, und ob Daten oder Programme behandelt werden. Führen alle drei Leitungen ein positives Signal, ist ein Interrupt aufgetreten.

Man kann beispielsweise durch ein Reset in den Supermodus gelangen, was besonders in Mehrbenutzersystemen von Interesse ist. Er ist aber auch über Exception-Vektor-Tabellen kontrolliert erreichbar, was besonders bei multitaskingfähigen Betriebssystemen von Bedeutung ist. Eine Exception, eine Ausnahme also, führt automatisch zum Supervisor-Modus. Ein Interrupt ist beispielsweise eine Exception, ebenso ein Trace. Welche Adresse beim Auftreten einer Exception angesprochen wird, enthält die Exception-Tabelle, die die ersten 1024 Byte eines Systems belegt. Für jeden Vektor ist dabei ein Langwort mit 32 Bit, reserviert. Auch die Interrupt-Prioritätsebenen sind in der Exception-Tabelle enthalten. Tritt ein Fehler, zum Beispiel eine Division durch Null, auf, aktiviert das die Vektornummer 5 mit der hexadezimalen Adresse 14 bis 17. Darauf folgt dann eine Fehlerbehandlungsroutine, genauer gesagt, eine Exception-Behandlung, die dem Anwender überlassen bleibt. Wie das System hier reagiert, hängt von der Programmierung ab. Bei den Interrupts (Vektornummern 19 bis 1F) liegen die Interruptprogramme von vornherein fest, bei anderen Exceptions sind sie applikationsabhängig. Auf alle Einzelheiten bei der Behandlung von Ausnahmefällen können wir hier nicht ausführlich eingehen. An dieser Stelle verweisen wir auf einschlägige Fachliteratur, die wir am Ende aufführen.

Nach der Beschreibung der internen Struktur des 68000er und den wichtigsten Anschlüssen wenden wir uns jetzt der Programmierung zu. Sie unterscheidet sich wesentlich von der anderer Mikroprozessoren. Es stehen nur 56 Grundbefehle zur Verfügung, was im Vergleich zu manchem 8-Bit-Prozessor sehr wenig erscheint. Da der 68000er aber über insgesamt 14 Adressierungsarten angesprochen werden kann und man noch zwischen den unterschiedlichen Daten unterscheiden muß, ergeben sich etwa 1000 Befehlskombinationen.

Die 14 Adressierungsarten lassen sich zunächst in sechs Hauptgruppen unterteilen:
1. Registerdirekt
2. Absolute Adressierung
3. Indirekte Adressierung
4. Implizierte Adressierung
5. Programmzählerbezogene Adressierung
6. Konstanten-Adressierung

Bei der ersten Gruppe gibt es zwei Formen, nämlich die direkte Daten- oder Adreß-Register-Adressierung. Bei der absoluten Adressierung wird zwischen »kurz« oder »lang« unterschieden. Die indirekte Adressierung kennt fünf Möglichkeiten:
3.1. Register indirekt
3.2. Register indirekt mit Postinkrement
3.3. Register indirekt mit Predekrement
3.4. Register indirekt mit Adreßdistanz
3.5. Register indirekt mit Index und Adreßdistanz

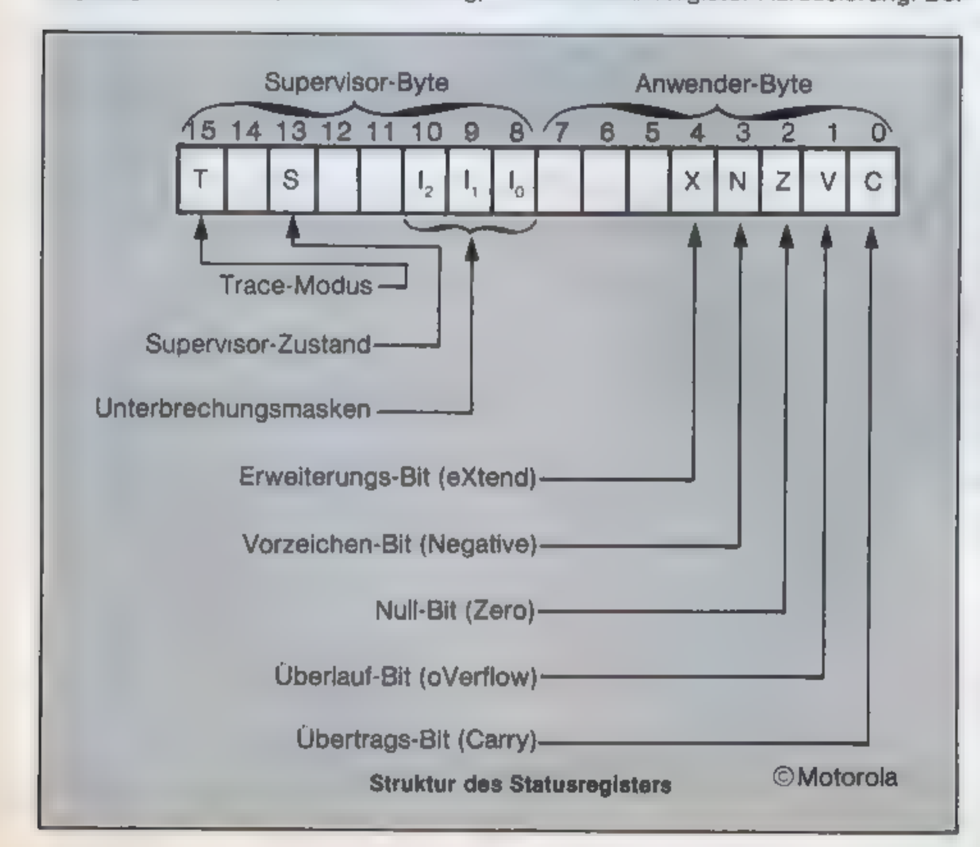
Programmzählerbezogen kann mit Adreßdistanz oder mit Index und Adreßdistanz gearbeitet werden. Bei der Konstanten-Adressierung schließlich wird zwischen Direktoperand und Byte-Konstante im Befehlswort unterschieden.

Allein diese Aufzählung zeigt schon, wie mächtig der 68000er zu programmieren ist. Durch einprägsame Mnemonics, das sind Abkürzungen für die einzelnen Befehle, fällt es auch dem Anfänger leicht, mit dem 68000er auf Assemblerebene umzugehen. Wer allerdings schon einige Erfahrungen mit anderen Mikroprozessoren gesammelt hat, muß umdenken. Nur wenige Spezifikationen lassen sich auf diesen Prozessor übertragen.

Auch noch erweiterbar

Der 68000er, mittlerweile auch als CMOS-Version verfügbar, kann mit den zahlreichen Peripherie-Bausteinen der 6000er Familie im Synchron-Betrieb eingesetzt werden. Nicht immer ist für den Einsatz dieses Prozessors ein spezieller Chip erforderlich. Soll aber ein Maximum an Leistung, wie es die Computer Atari ST und Commodore Amiga bieten, erreicht werden, führt kein Weg an der Entwicklung von Chips vorbei. Denn sie nehmen dem Prozessor ein Großteil der Arbeit ab, für die er nicht konzipiert wurde, die er in vielen Systemen aber erfüllen muß.

Seit über sechs Jahren gibt es jetzt den 68000er, der weiter verbessert wird. Immer schnellere Typen werden angekündigt. Auch eine Version, bei der alle 32 Bit der Adreßregister herausgeführt sind, soll in absehbarer Zeit verfügbar sein. Doch zur Zeit haben die Computerentwickler noch kein System vorgestellt, das auch nur annähernd die derzeitig verfügbaren 68000er voll nutzt. (rs)



Struktur des Statusregisters

Weiterführende Literatur
»Das Prozessorbuch zum 68000«, Grohmann, Eichler Data Becker, Düsseldorf. ISBN: 3-89011-094-0
»Der Mikroprozessor 68000«, Coffron, Vieweg Verlag, Braunschweig, ISBN. 3-528-04330-X
»Programmierung des 68000«, Vieillefond, Sybex-Verlag, Düsseldorf ISBN: 3-88745-080-4
»68000 Mikroprozessorhandbuch«. Kane, McGraw Hill, Hamburg ISBN. 3-89028-030-7
»M68000 Familie«, Hilf, Nausch, Te-wi-Verlag, München, ISBN 3-921803-16-0

Furore

Viel Aufruhr verursachte Atari mit der neuen ST-Serie. Und das mit Recht.

Jack Tramiel sorgte für viel Aufregung in der Computerszene durch den Kauf von Atari. Noch mehr aber erregten die neuen Atari-Computer die Gemüter.

Die neue Computergeneration von Atari birgt als »Herz« den Mikroprozessor 68000 von Motorola. Die daraus resultierende Leistungsfähigkeit blieb in dieser Preisklasse bisher unerreicht. Die Domäne der 16/32-Bit-Prozessoren war den Heimanwendern vorenthalten. Atari beschreitet aber nicht nur in der Hardware neue Wege, auch in der Software hat sich ein ebenso gewaltiger Schritt nach vorne vollzogen: Der Atari ST bietet seinem Anwender eine Benutzeroberfläche, die die Bedienung eines Computers zu einem Kinderspiel macht. Keine abstrakten Kommandos sind mehr gefragt, sondern man »zeigt« dem Computer, was er tun soll.

Dieser Bedienungskomfort basiert auf »GEM«. Der »Graphics Environment Manager« ist eine Schnittstelle zwischen dem Betriebssystem und dem Benutzer. Sie stellt dem Anwender Symbole, Pull-Down-Menüs und Fenster zur Verfügung. Diese Symbole sind zum Beispiel ein Karteikasten oder ein Papierkorb. Um sich das Inhaltsverzeichnis anzuschauen, steuert man den Mauszeiger auf die Diskette und drückt zweimal kurz hintereinander die linke Maustaste. Schon erscheint in einem Bildschirmausschnitt das Inhaltsverzeichnis, dargestellt aus kleinen Symbolen. Möchte man eine Datei löschen, zeigt man mit dem Mauscursor auf das betreffende Symbol und drückt die linke Maustaste. Dadurch färbt es sich schwarz. Behält man die Maustaste gedrückt, so läßt sich das Symbol verschieben. Steuert man es zum Papierkorb, löscht es die Datei.

Die Piktogramme stellen die Einrichtungsgegenstände eines Büros und ihre Funktionen dar. Das Hauptmenü des GEM heißt auch »Desktop«, also Schreibtisch. Darauf basiert die Idee dieser komfortablen Benutzerschnittstellen. Der Anwender befaßt sich mit Bildern, die er bereits kennt, deren Funktionen ihm geläufig sind. Die Diskette ist ein Karteikasten. Das Löschen entspricht dem Hineinwerfen in den Papierkorb.

Katz und Maus

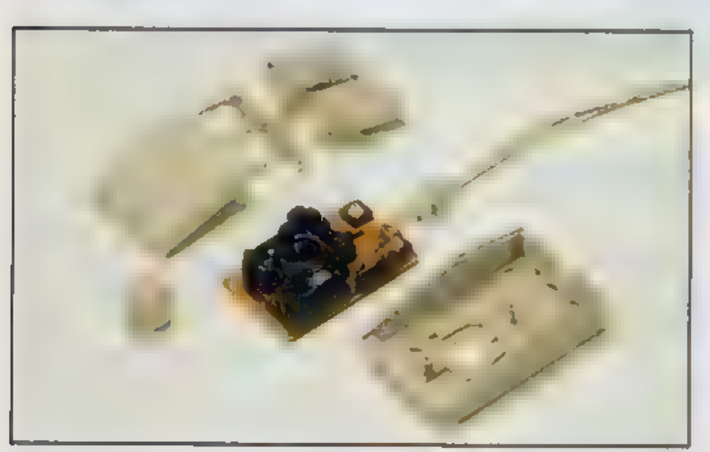
Die Bedienung der Benutzeroberfläche erfolgt mit der Maus, deren Hauptbestandteil eine 2,5 Zentimeter große Metallkugel ist. Das hohe Eigengewicht von 45 Gramm und ein Gummiüberzug verhindern ein »Durchrutschen« auf glatter Fläche.

An der Unterseite des 98x62x38 Millimeter großen Gehäuses befindet sich ein 1,6 Zentimeter großes Loch, durch das die Kugel 2 Millimeter herauslugt. Schiebt man die Maus über den Schreibtisch, dann dreht sich die Kugel und bewegt damit drei Walzen. Zwei davon sind im 90-Grad-Winkel zueinander angeordnet und übertragen die Bewegung in Richtung der X- und Y-Achse an den Computer. Die dritte Walze drückt die Kugel leicht gegen die beiden anderen. Dieses Prinzip gewährleistet eine gleichmäßige Übertragung der Bewegung.

Die beiden Meß-Walzen besitzen an ihren Achsen kleine Scheiben mit Löchern und zwei optischen Signalgebern, die jede Drehung der Scheiben registrieren. Die Auflösung beträgt vier Schritte pro Millimeter. Sie reagieren dadurch auf kleinste Bewegungen. Das wird eindrucksvoll demonstriert. Bei Programmen, die mit Maus- und Tasteneingaben arbeiten, verschwindet der Mauscursor, wenn er für eine kurze Zeitspanne nicht bewegt wird. Es ist aber nahezu unmöglich, die Hand auf die Maus zu legen, ohne daß es registriert wird und der Mauscursor wieder erscheint. Er läßt sich dadurch sehr präzise und punktgenau steuern. Die schnellste Bewegung beträgt dabei 250 Millimeter in der Sekunde. Selbst bei sehr schnellen Bewegungen bleibt die Maus »am Ball«. Probleme gibt es da eher mit der Mechanik (genauer: mit der Rollkugel), die durch ihr Eigengewicht von 45 Gramm relativ schwer ist und auf einer glatten Unterlage durch die eigene Trägheit leicht rutschen kann. Bei normaler Benutzung funktioniert sie allerdings vortrefflich.

Am vorderen Teil des Gehäuses sind zwar zwei Tasten angebracht, die linke Maustaste spielt dabei aber die Hauptrolle. »GEM«-Funktionen ruft man fast ausschließlich durch ein- oder zweimaliges Drücken auf diese Taste ab (»Klicken«). Die darunter verborgenen Tast-Schalter sind sehr klein und haben deshalb einen kurzen Hub von nur etwa einem Millimeter.

Die Datenübertragung findet über ein 90 Zentimeter langes Kabel statt, das die Bewegungsfreiheit der Maus nicht einschränkt. Durch eine abnehmbare Plastikscheibe, in der sich das Loch für die Kugel befindet, läßt sie sich aus dem Gehäuse nehmen. Das ist wichtig, da sie durch vieles Rollen über den Schreibtisch leicht verschmutzt. Reinigen darf man sie nur trocken, da sonst die Gummioberfäche angegriffen wird. Die drei Rollen sind relativ schmutzempfindlich. Deshalb kann man auch sie leicht zur Reinigung aus dem Gehäuse entfernen.



▲ Klein, aber oho! Die Maus in ihren Einzelteilen.

Das Diskettenlaufwerk. Die kleine Steckerplatine ist über Kabel mit dem Laufwerk verbunden. ▼

Atari bietet für den ST zwei Diskettenlaufwerke mit 3½-Zoll- Format an. Das SF 354 beschreibt die Disketten einseitig und erreicht eine Speicherkapazität von 500 KByte (unformatiert). Formatiert bleiben noch 360 KByte. Die Diskettenstation SF 314 nutzt beide Seiten einer Diskette und kann deshalb mit den doppelten Werten aufwarten.

Fleißiges Lieschen

Die Abmessungen des grauen Gehäuses betragen 237x137x60 Millimeter. Dadurch benötigt es auf dem Schreibtisch nicht viel Platz. Platzraubender ist das zusätzliche Netzteil.

Das Gehäuse des Laufwerks hat auf der Rückseite rechts den Ein-/Ausschalter, links daneben eine Steckerbuchse für das Verbindungskabel zum Netzteil. Daneben findet man noch zwei gleich aussehende Buchsen. Eine davon benötigt der Stecker des Verbindungskabels zum Computer, die zweite ist als Anschluß eines weiteren Laufwerks gedacht.

Öffnet man das Gehäuse, so sieht man, daß die drei Buchsen und der Schalter auf einer kleinen Platine sitzen, die ein Flachbandkabel und vier dünne Drähte mit dem eigentlichen Laufwerk verbinden.

Das Laufwerk stammt von Epson. Seine Disketten beschreibt der Atari ST auf einer Seite mit 9 Sektoren je Spur. Mit einer Sektorengröße von 512 Byte ergeben sich durch die 80 Spuren 360 Kbyte. Für die Laufwerke der kleinen »Brüder« des Atari ST fehlt leider eine Anschlußmöglichkeit.

Fliegende Untertasse

Die Disketten im 3½-Zoll-Format sind zwar teuerer als Speicherscheiben im etablierten 5¼-Zoll-Format. Sie bieten aber einige Vorteile. Die stabilere Plastikhülle schützt sie besser vor Beschädigung. Zieht man die Diskette aus dem Laufwerk, hat sich bereits ein Metallschieber über das Schreib/Leseloch geschoben, und bewahrt sie so vor unabsichtlichen »Fingerabdrücken«. Eine Diskettenhülle erübrigt sich damit.

Diese Vorteile muß man mit einem Nachteil bezahlen. Die Disketten lassen sich nicht wenden. Bei 5¼-Zoll-Disketten »überlistet« man das einseitige Laufwerk durch einen Trick. Man zwickt einfach gegenüber dem Schreibschutzloch ein zweites und schiebt die Diskette mit der anderen Seite nach oben in das Laufwerk, und schon ist die zweite Seite fertig zum Beschreiben. Da beide Seiten der Diskette identisch sind, klappt das ohne Probleme. Die kleineren Speicherscheiben des ST verhindern das durch die Antriebsplatte an der Unterseite und den Metallschieber. Ein doppelseitig schreibendes Laufwerk ist also empfehlenswert. Die Mehrausgabe hat sich bei den Diskettenpreisen schnell amortisiert.

Bei den Monitoren kann man unter dem monochromen SM 124 und dem Farbmonitor SM 1224 wählen.

Der Schwarz/Weiß-Monitor zeichnet sich durch eine hervorragende Darstellung mit einer Höchstauflösung von 640x400 Punkten aus. Das beschränkt sich aber ausschließlich auf diesen Modus. Der ST prüft beim Bootvorgang das Signal am obersten Pin der Monitorbuchse. Ein High-Signal signalisiert den monochromen Monitor, und der ST arbeitet ausschließlich im hochauflösenden Modus. Versucht man im Einstellmenü die Auflösung zu reduzieren, so ignoriert der ST diesen Versuch einfach.

Der Farbmonitor erlaubt zwei Auflösungen 320x200 und 640x200. Die höchste Auflösung bleibt dem monochromen Monitor vorbehalten. Aber auch die Darstellung des Farbmonitors kann sich sehen lassen. Sie behält ihre Schärfe auch in den Ecken bei; das kann nicht jeder Farbmonitor von sich behaupten. Auch geringe Farbabstufungen lassen sich präzis unterscheiden. Genauso wie beim monochromen Monitor registriert der ST sofort die Art des Monitors und schaltet dann automatisch in den Farbmodus. Der Computer arbeitet nach dem Start im 320x200-Modus. In den 640x200-Modus muß man ihn manuell umschalten. Beachten sollte man, daß der ST dazu naturlich den Bildschirm löscht. Eventuell benötigte Grafiken deshalb vorher sichern!

Guckloch in die Seele

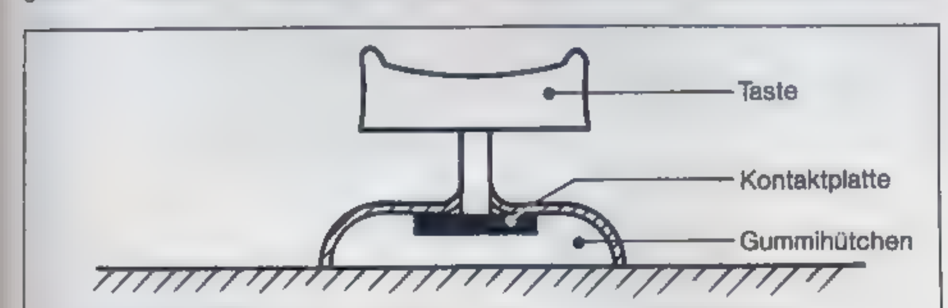
Die Gehäuse beider Monitore präsentieren sich natürlich auch im grauen, unauffälligen Atari-Look. Der Computer selbst hat eine integrierte Tastatur.

Durch diese Symbiose spart man zwar Platz, aber die Tastatur läßt sich der Schreibhaltung nicht anpassen. Separate Tastaturen bieten mehr Komfort, indem sich aus deren Gehäuseboden meist kleine Füßchen ausklappen lassen. Damit kann man die Neigung verstellen. In unserem Test zeigte sich aber, daß auch der ST eine relativ ermüdungsfreie Arbeitshaltung erlaubt.

Das Innenleben

Die Tastatur des Atari ST besteht aus insgesamt 94 Tasten. Wer gewohnt ist, mit einer Schreibmaschine zu arbeiten, den stört der zu weiche Druckpunkt, ansonsten haben wir keine Beanstandungen.

Die Anordnung unterteilt sich in vier Blöcke. Der Hauptblock ist nach links versetzt und enthält die Buchstaben-, Zahlen-, SHIFT-Tasten sowie die DELETE-, RETURN-, Backspace- und große Leer-Taste. Darüber sitzt der Funktionstastenblock. Durch die schräge Anordnung der Tasten harmonieren sie rein optisch gut mit den schräg verlaufenden Kühlrippen. Rechts vom Hauptblock befinden sich die Cursor-, HELP-, UNDO-, INSERT- und CLR-HOME-Tasten, rechts davon ein Zahlenblock.



Der mechanische Aufbau einer Taste

Die Mechanik ist einfach, aber effektiv: unter dem Tastenknopf sitzt ein Gummihütchen. In dessen Mitte bildet ein kleines elastisches, stromleitendes Plättchen den Kontaktgeber.

Drückt man eine Taste, so preßt man dadurch das Gummihütchen nieder, und das Plättchen verbindet zwei Leiterbahnen, der Kontakt ist geschlossen. Ein Nachlassen der Elastizität der Gummihütchen ist nicht zu erwarten. Ein bekannter Tastatur-Hersteller bestätigte auf unsere Anfrage mindestens 15 Millionen einwandfreie Tastendrucke einer solchen Gummi-Mechanik. Sechzehn dünne Kabel verbinden die Tastaturplatine mit der Hauptplatine.

Für das Dekodieren von Tastatur, Eingaben der Maus und der Joysticks sitzt unter der Tastaturplatine ein Mikroprozessor: der HD6301V1, ein 8-Bit-Prozessor mit 1 MHz Taktfrequenz. Er arbeitet mit einem der beiden MC6850 Bausteine auf der Hauptplatine zusammen.

Ein Abschirmblech unter der Tastatur benötigt der Atari ST zur Funkentstörung. Da die Postbestimmungen dazu immer strenger werden, umhüllt ein zweites die Hauptplatine. Damit ist gewährleistet, daß sich keine Störsignale »herausschleichen«.

Die Hauptplatine hat eine Größe von 445x240 Millimeter. Ihre relativ wenigen Bauteile verwundern bei einem Computer dieser Leistungsklasse. Das ist jedoch kein Nachteil, denn je weniger Bauteile auf der Platine, desto weniger können kaputtgehen.

Auf der linken Seite stecken nur zwei Bausteine in einer Reihe von Sockeln, die eigentlich sechs aufnehmen könnten. Die fehlenden vier Bausteine hier sollen einmal die restlichen Betriebssystemfunktionen aufnehmen. Zur Zeit bergen die zwei ROM-ICs nur die Bootfunktionen und eine Systemmeldung. Sie erscheint, wenn sich keine oder eine fehlerhafte Diskette im Laufwerk befindet.

Ursprünglich war geplant, den Atari ST mit 192 KByte ROM auszuliefern. Es sollte Betriebsprogramm und GEM enthalten. Aufgrund von Fehlern im Betriebssystem wurde davon abgesehen. Man kann dies mit einem lachenden und einem weinenden Auge betrachten. Befindet sich das Betriebssystem und GEM im ROM, dann belegt es keinen Speicherplatz im RAM-Bereich (oder nur soviel, wie es für die Systemvariablen benötigt). Ändert sich andererseits aber etwas im ROM, so ist es schwierig, diese Änderung an die Kunden weiterzugeben. Treten jetzt noch Änderungen auf, sendet man einfach eine neue Version an die Händler, und der Kunde kann sie sich kopieren.

Darüber sitzt der ROM-Port, der ROMs mit einer Speicherkapazität von 128 KByte aufnehmen kann. Sie enthalten bei Bedarf Erweiterungen für das Betriebssystem oder Programme. Dieser ROM-Schacht ist nicht nur ein Andenken an frühere Zeiten, als Atari am Spielemarkt große Erfolge verzeichnete. Der 68000-Prozessor hat einen Adreßbereich von 16 MByte. Angesteckte ROM-Module zwacken nichts vom Speicherplatz des RAM ab, die Ladezeiten sind bekanntlich unerheblich. Probleme mit der Datensicherung, wie man sie von Disketten kennt, gibt es nicht. Oft benötigte Programme wären also sehr gut auf einem ROM aufgehoben, was daran scheitert, daß zur Zeit allerdings noch niemand Software auf ROM anbietet.

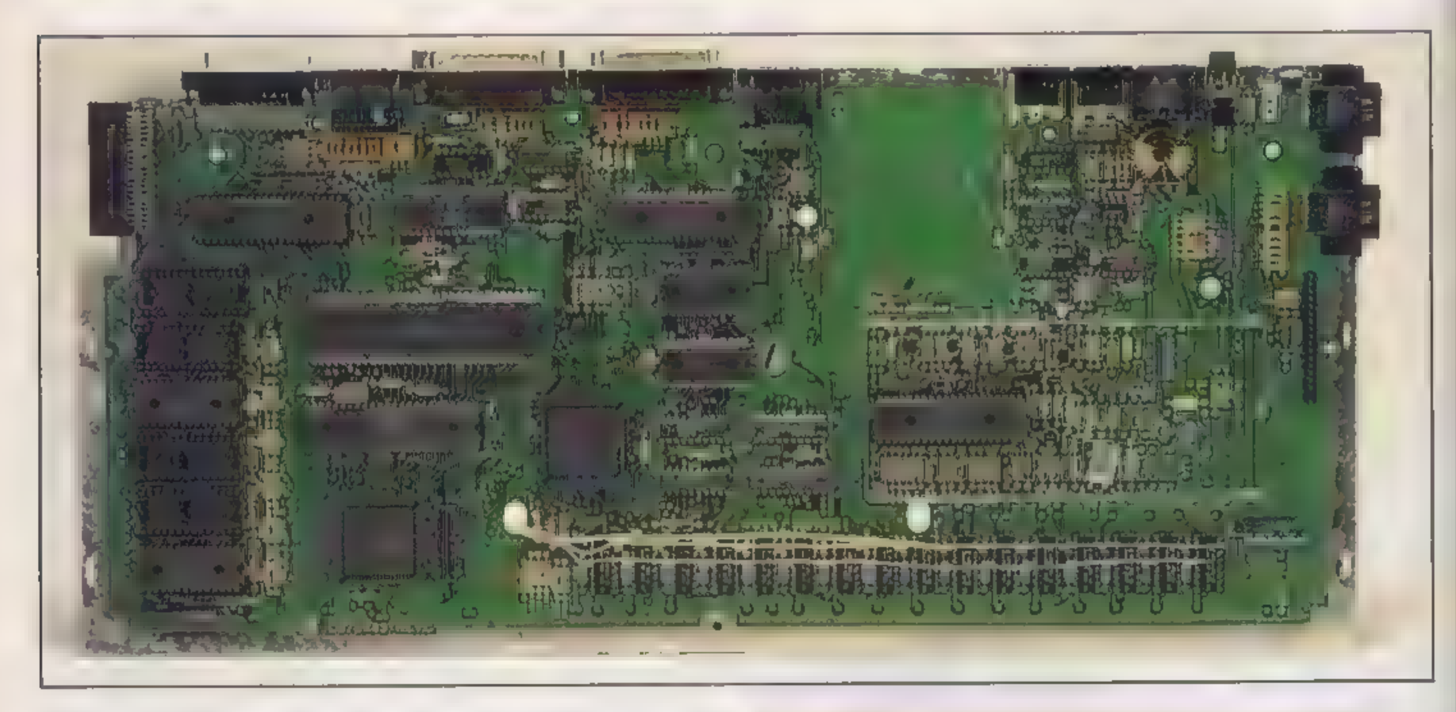
Rechts vom ROM sitzt der DMA-Controller. Er ist in der Lage, die RAMs direkt mit externen Daten zu versorgen, ohne die Leistungskapazität des 68000-Mikroprozessors zu beanspruchen. Die Übertragungsrate zwischen Diskettenlaufwerken oder Festplatte findet durch diesen Baustein mit 1,3 Megabyte pro Sekunde statt. Auf der Systems in München konnte man seine hohe Übertragungsgeschwindigkeit in Zusammenarbeit mit einer Festplatte bewundern. Dieser direkte Speicherzugriff, ohne Mithilfe der CPU, blieb bisher Computern höherer Preisklassen vorbehalten.

Der DMA-Controller verwaltet auch zwei Kanäle für die MIDI-Schnittstelle. Mit einer Transfergeschwindigkeit von 31,25 KByte pro Sekunde sind sie weit langsamer als die acht weiteren Kanäle. Diese schicken die Daten mit 1,3 MByte pro Sekunde durch die Leitung. Dieser schnelle Transfer ist für die Festplatte oder ein Netzwerk, das Atari in Planung hat, vorgesehen.

Die Datenübertragung wickelt der DMA-Controller über ein DMA-Mode-Control-Register und einen 32-Byte-FIFO-Speicherzugriff ab. FIFO ist die Abkürzung für »First In First Out« und heißt »Was zuerst hereinkommt, geht zuerst raus«. Also genau umgekehrt, wie man es von einem Stack kennt.

DMA für schnellen Zugriff

Der Disketten-Controller befindet sich neben dem DMA-Controller. Der Baustein trägt die Bezeichnung »WD 1770/1772«. Er ist in der Lage, zwei Diskettenlaufwerke und ein Festplattenlaufwerk zu steuern. Das erstaunt, denn normalerweise kann ein solcher Controller bedeutend mehr Laufwerke kontrollieren. Auf unsere Nachfrage hin sagte Atari-Chefentwickler Shiraz Shivji: »Natürlich kann er das, aber wir haben nicht genügend Pins für mehr Anschlüsse frei«. So begnügt man sich eben mit zwei Diskettenstationen. Wer mehr benötigt, kann auf eine RAM-Disk oder eine Festplatte – in Planung – zurückgreifen. Der Laufwerk-Controller arbeitet bei dem Hin- und Herschaufeln



Die Hauptplatine des Atari 520 ST+. Auf die 16 RAM-Bausteine wurden einfach weitere 16 »huckepack« draufgesetzt und angelötet.

von Daten natürlich mit dem DMA-Controller zusammen

Unter dem Disketten-Controller schlägt das Herz dieses Computersystems: der Mikroprozessor 68000 von Motorola. 1977 begann Motorola mit seiner Entwicklung. Auf den Markt kam er 1979. Es gelang Motorola zum ersten Mal, einen ähnlichen Befehlssatz auf einem Baustein unterzubringen, wie ihn bis dahin nur Mini-Computer besaßen.

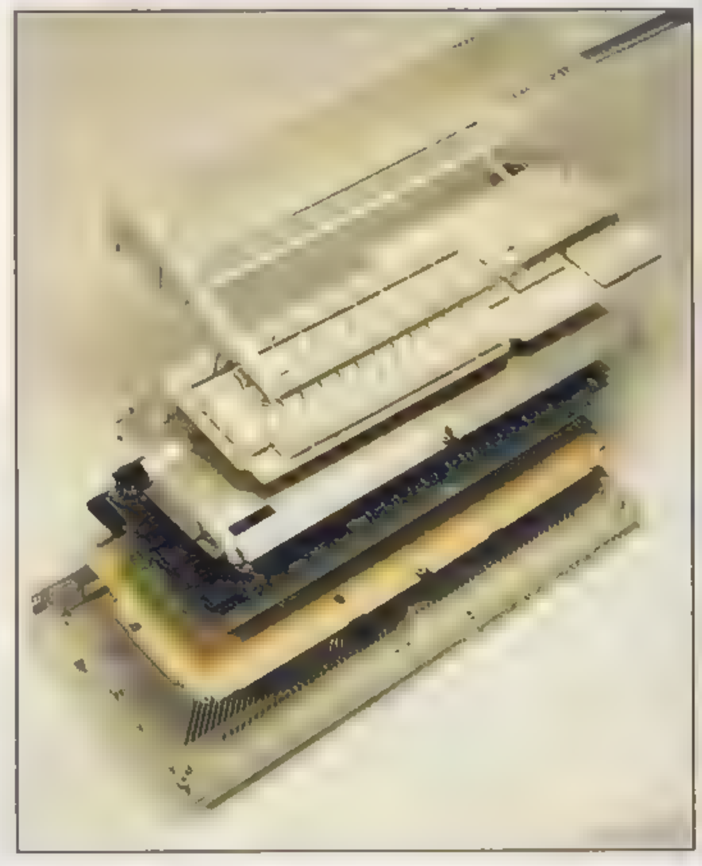
Softwareentwicklern hat dieser Baustein einiges zu bieten: acht 32-Bit-Datenregister, sieben 32-Bit-Adreßregister, zwei 32-Bit-Stackpointer, einen 32-Bit-Programmcounter und ein 16-Bit-Statusregister. Sein Adressierbereich umfaßt durch den 24-Bit-Adreßbus 16 MByte. Durch sogenannte »Traps« läßt sich Multitasking einfacher programmieren als bei anderen Prozessoren. Die Verwendung der schnellen und flexiblen Programmiersprache »C« unterstützt der 68000 optimal. Spezielle Fehlerbehandlung machen das Programmieren von Software einfach und schnell. Obwohl sein Maschinencode-Befehlssatz nur 56 Befehle umfaßt, ergeben sich durch die Kombination mit 14 Adressierungsarten annähernd 1000 Befehle. Da bleiben keine Wünsche offen!

Im ST »werkelt« die 8-Megahertz-Ausführung des 68000er.

Atari verwendet damit einen Baustein, der ausgereift ist und dennoch neueste Prozessortechnologie darstellt.

Unter diesem »Kraftpaket« sitzt der 68901, ein multifunktionaler Chip, der unter anderem für die Interruptsteuerung und den Timer zuständig ist. Eines seiner Haupteinsatzgebiete ist die Steuerung der Centronics-Schnittstelle. Er arbeitet dabei mit dem Sound-Chip zusammen und benutzt dessen I/O-Port B. Seine Daten sendet er in der altbekannten 8-Bit-Breite mit einer Übertragungsrate von 32 KBaud.

Auch die serielle Schnittstelle schickt ihre Daten mit Hilfe dieses ICs an die Außenwelt. Dabei hilft wieder der Soundbaustein, diesmal mit seinem Port A, aus. Das Sende- und Empfangstempo kann zwischen 50 und 19200 Baud liegen. Die »Memory Management Unit« verwaltet bis zu 4 MByte. Atari deutete an, eine weitere ST-Version auf den Markt zu bringen und mit zwei MMU-Bausteinen auszustatten. Sie ist dann in der Lage, 8 MByte direkt zu adressieren.



Der Atari ST »aufgefächert«. Deutlich erkennt man die einzelnen Komponenten.

Dieser Baustein dient im Atari ST zur Adreßdecodierung. Sie kontrolliert und steuert jeden Zugriff auf die RAM-Bausteine. Die GLUE wurde von Atari speziell für die ST-Serie entwickelt.

Die beiden Atari-ST-Versionen, der 260 ST, der 520 ST und der 520 ST+, unterscheiden sich durch die verfügbare Speicherkapazität. Ist der 520 ST+ mit 1 MByte RAM ein wahrer Spei-

Die Rückseite des ST mit den Schnittstellen

cherprotz, so muß sich der 260 ST und der 520 ST mit der immer noch stattlichen Hälfte begnügen. Zur CeBit 86 darf man sich allerdings auf die ROM-Version des Betriebssystems freuen. Dadurch gewinnt man wieder einigen Speicherplatz hinzu, den jetzt noch das Betriebssystem benötigt. Von Atari ist geplant, diese sechs Bausteine für 99 Mark auszuliefern.

Auf der Platine sitzen 16 Stück 256-KBit-Bausteine, die 512 KByte RAM bieten. Beim 520 ST+ wurde auf jeden der 16 ICs einfach noch einmal ein Baustein »huckepack« draufgesetzt und angelötet. Genau nach dem Verfahren, das in Happy-Computer 1/86 beschrieben wurde. Das Aufrüsten von einem 512 KByte ST stellt einen lötkolbenerfahrenen Bastler vor keine Probleme. Allerdings entfällt durch einen solchen Eingriff natürlich die Garantie.

Über den RAMs tut der Video-Chip sein Werk. Er sitzt unter einer Abschirmung aus Metall, damit er seine Hochfrequenz-Signale nicht unkontrolliert streuen kann.

Dieser »Shifter« setzt den Inhalt des Bildschirmspeichers in ein Signal um, das der Monitor verwerten kann.

Den »Shifter« entwickelte Atari auch höchstselbst.

Links darüber befinden sich zwei 6850 ACIA-Bausteine, sogenannte Asynchrone-Interface-Bausteine, die Peripheriegeräte steuern.

Darf's ein bißchen mehr sein?

Einer davon empfängt die Eingaben der Tastatur. Seine Datenübertragungsrate beträgt 7812,5 Baud. Er arbeitet dazu mit dem HD6301V1-Baustein zusammen, der auf der Unterseite der Tastaturplatine sitzt.

Der zweite Baustein mit einer Übertragungsrate von 31,25 KBaud kontrolliert die MIDI-Schnittstelle.

Für die Tonerzeugung griffen Cheftechniker Shiraz Shivji und seine Ingenieure auf den Sound-Chip AY-3-8910 von Yamaha zurück. Leider überzeugen die damit erreichten Tonqualitäten nicht sehr.

Dieser Baustein ist zusätzlich für die akustische Rückmeldung bei jedem Tastaturdruck zuständig. Dieses Softwareverfahren bietet einen großen Vorteil gegenüber der Hardwarelösung: Man kann den Tastaturklick abschalten.

Der Baustein arbeitet in einem Frequenzbereich von 30 Hz bis 125 KHz. Er verfügt über 3 Kanäle und einen Rauschgenerator. Der Ton kommt aus einem Lautsprecher im Monitor.

Aber noch bei mehr Dingen hat dieser Baustein seine Bits im Spiel. Seine interne Uhr synchronisiert die Datenübertragung der Centronics-Schnittstelle.

Der Atari ST kann seine Bildschirmausgabe in drei verschiedenen Auflösungen darstellen. Die niedrigste mit 320x200, die mittlere mit 640x200 nur mit dem Farbmonitor, die maximale Auflösung von 640x400 Punkten die ausschließlich dem monochromen Monitor überlassen bleibt.

Die Synchronisation von 50 Hz für Europa oder 60 Hz für USA legt der Atari über das Betriebssystem im Sync-Mode-Register fest.

Um die Farbwerte der Bildschirmpunkte zu speichern, liegen hinter dem Bildschirmspeicher Bit-Planes. Je nach Höhe der Auflösung sind das 4, 2 oder 1 Bit-Plane.

Über die Rot-, Grün-, Blauwerte des Farbsignals erzeugt der Atari ST seine Farbpalette. Sie setzt sich aus jeweils acht Abstufungen zusammen, die 512 Farben (8 x 8 x 8 = 512) ergeben. In 16 Registern sind die aktuellen Werte gespeichert. Mehr als 16 Farben gleichzeitig kann der ST nicht darstellen. In der Kombination mit einem guten Farbmonitor beeindrucken aber seine Grafiken, wie bereits bewiesen.

Atari bietet den ST in drei Versionen an. Den 260 ST mit 512 KByte RAM und einem Scart-Kabel, um ihn an einem Fernsehgerät mit Scart-Eingang und Composite-Video-Signal zu betreiben. Leider weisen nicht viele Fernseher einen solchen Eingang auf. Zu diesem Einsteigerpaket ist die Maus zu empfehlen. Im Handbuch weist Atari auch auf die Steuerung des Mauszeigers durch die Cursortasten hin. Wer es einmal probiert hat, stellt sehr schnell fest, daß damit kein vernünftiges Arbeiten mit der Benutzeroberfläche möglich ist. Die beiden nächsten Versionen sind nur als »Paket« zu haben. Das heißt, man muß einen Monochrom-Monitor, die Maus und ein einseitiges Diskettenlaufwerk dazu kaufen. Beide Geräte unterscheiden sich nur durch die RAM-Kapazität. Der 520 ST hat 512 KByte, sein großer Bruder mit dem Pluszeichen hinter dem T, 1 MByte. Er ist damit der Mega-Atari.

Allen Geräten liegen Disketten bei. Ohne Diskettenlaufwerk hilft also der 260 ST nichts! Dem 260 ST und 520 ST+ sind zwei solche »Speicherscheiben« beigegeben. Sie enthalten das Betriebssystem, die Benutzeroberfläche GEM und zwei Programmiersprachen: Logo und Basic. Der Käufer eines 520 ST (ohne Plus) bekommt noch »First Word« und »DB-Master«. Das erste ist ein Textverarbeitungsprogramm mit GEM-Funktionen. Das zweite eine Datenbank, die dBase zwar noch etwas nachsteht, aber auch einiges zu bieten hat. Die beiden Programme sollen die RAM-Differenz von 512 KByte zwischen dem 520 ST und dem 520 ST+ ausgleichen.

Ohren anlegen

Die Softwarefirmen, besonders viele inländische, stehen hinter diesem neuen Computer. Für den ST steht soviel Software bereit, wie es noch nie so kurze Zeit nach einer Einführung der Fall war.

Da ist ein anderer Punkt, der es Atari sehr schwer macht, in den professionellen Markt einzusteigen. Der Atari ist zu keinem anderen Computer kompatibel. Die Finanzen sind noch nicht so gefestigt, daß man langfristig rechnen kann – obwohl das zugegebenermaßen in diesem Markt bei kaum einer Firma der Fall ist.

Leider haben sich die Hersteller von Computern mit 68000er-Mikroprozessor noch nicht auf ein einheitliches Betriebssystem geeinigt. Was auf dem einen Computer läuft, ist auf einem anderen Computer noch lange nicht lauffähig.

Die bereits erhältlichen Software-Emulatoren täuschen lediglich eine Kompatibilität vor. Die gleichen Ablaufgeschwindigkeiten wie unter dem Betriebssystem lassen sich nicht erreichen.

Bisher haben zwar viele Softwarehäuser für den Atari ST gute und leistungsstarke Programme entwickelt. Die Großen in dieser Branche ließen aber bisher die Finger von diesem Newcomer.

Nun, so kann man gespannt sein, wie Atari in den Profimarkt einsteigt.

Im Heimcomputermarkt spielt der ST bereits eine wichtige Rolle und wird sie bestimmt nicht so schnell aus der Hand geben. (hb)

Wolf im Schafspelz?

Kaum ein anderer Computer hat die Gemüter bereits im Stadium eines Gerüchts so erhitzt wie der Amiga. Kaum ein anderer hat so viele Vorschußlorbeeren geerntet. Jetzt ist er da!

Wie kein anderer Computer vor ihm, wurde der Commodore-Amiga schon lange vor seinem wirklichen Erscheinen von den wildesten Gerüchten begleitet. Die ersten tauchten Winter 1983 auf der Comdex in den USA auf. »Amiga« war damals noch der Name einer Firma, die nur Joysticks – sehr gute allerdings – verkaufte. Die Amiga-Joysticks gibt es noch heute. Einen Personal Computer zu bauen und noch dazu einen, der alles bisher dagewesene in den Schatten stellen sollte, traute dieser Firma eigentlich kaum jemand zu. Als man jedoch den Namen eines der Männer hörte, die am Design des Amiga mitarbeiten sollten, wurden die Zweifel etwas geringer. Jay Miner, der schon die hochgelobten Grafik- und Sound-Chips für den Atari 400/800 entwickelt hatte, sollte dem Amiga dieselben Dienste leisten. Diese Chips sollten einem Heimcomputer Fähigkeiten verleihen, die sonst nur in speziellen Grafikcomputern für zig-Tausend Mark zu finden sind.

Im Sommer 1985 wurde der Amiga dann endlich im Rahmen einer millionenschweren Multimedia-Show in New York vorgestellt. (Seine deutsche Premiere steht auf der CeBIT in Hannover im März 1986 bevor). Kaum einen der Anwesenden dürfte die New Yorker Vorstellung kalt gelassen haben. Ganz abgesehen von reichlich Prominenz, mit der sich der Amiga schmückte, so zum Beispiel dem schon etwas betagten Pop-Künstler Andy Warhol und der Rocksängerin Debby Harrie (Blondie), wurden vor allem die Grafik-Fähigkeiten des Amiga groß herausgestellt.

In diesen fast drei Jahren, zwischen Entwicklungsbeginn und Vorstellung, kursierten die wildesten Gerüchte über eine wahre Supermaschine in der Presse und den Computernetzwerken. Einmal wurden die Gerüchte sogar Amiga selbst zu bunt, worauf die Firma ein Papier veröffentlichte, das eine ungefähre Beschreibung der Maschine enthielt. Nun, da der Amiga auch die alte Welt erreicht hat, stellt sich die Frage, ob sich das Warten gelohnt und die Gerüchteküche zurecht so heftig gekocht hat. Im folgenden soll eine ausführliche Beschreibung die Fähigkeiten

und Grenzen des Amiga aufzeigen, damit sich der Leser selbst ein Urteil bilden kann.

Unsere Meinung dazu vorneweg: Es hat sich gelohnt. Es gibt auf vielen Gebieten im Moment unter 20000 Mark wohl keinen Computer auf dem Markt, der dem Amiga das Wasser reichen kann.

Personal Computer-Look

Äußerlich macht der Amiga, »Amiga 1000« um genau zu sein, (Commodore hat offensichtlich noch weitere Modelle in Planung) einen sehr attraktiven Eindruck. Das »Basispaket«, das natürlich ausbaufähig ist, besteht aus insgesamt vier Komponenten:
- einer Haupteinheit
- einer separaten Tastatur
- einem Farbmonitor
- einer Maus.

Man kann (und sollte auch!) noch ein zusätzliches Diskettenlaufwerk erwerben, das außen an die Haupteinheit angeschlossen wird.

An einem langen und flexiblen Spiralkabel ist die Tastatur mit der Haupteinheit verbunden. Die Maus findet über ein ebenfalls langes Kabel an der rechten Gehäuseseite an die Haupteinheit Anschluß. Der Bildschirm hat seinen Platz auf der Haupteinheit, das zusätzliche Diskettenlaufwerk rechts davon – leider durch ein kurzes, starres Kabel in der Bewegungsfreiheit etwas gehandicapt. Es paßt aber auch sehr gut hochkant rechts neben den Bildschirm auf die Haupteinheit.

Die Tastatur ist angenehm flach und ist mit 89 Tasten ausgestattet. Sie besitzt neben dem normalen Satz Schreibmaschinen-Tasten, einen separaten Zehnerblock für die schnelle und komfortable Zahleneingabe, einen zweiten Block mit den zehn Cursor-Tasten und eine Reihe von Funktionstasten am oberen Rand.

Links und rechts neben der Leertaste liegen zwei besondere Tasten, die verschiedene Formen des Amiga-Symbols tragen. Es sind (in Kombination mit anderen Tasten) spezielle Funktionstasten, wie sie heute auch bei vielen anderen Rechnern zu finden sind. Direkt neben den Amiga-Tasten befindet sich auf beiden Seiten noch eine sogenannte »Alt-Taste«. Hält man die Alt-Taste fest und drückt dann einen »normalen« Buchstaben, so erhält man Sonderzeichen. Fast jede Buchstabentaste ist mit einem solchen alternativen Symbol versehen, weshalb der Amiga wohl mit Recht als ein internationaler Computer zu bezeichnen ist.

Mit zwei kleinen Beinchen am hinteren Rand kann die Tastatur bei Bedarf schräg gestellt werden und dadurch eine Sitzhaltung begünstigt, die viele Benutzer beim Tippen als bequemer empfinden. Der Anschlag der Tasten ist leicht und angenehm, sowohl für diejenigen, die nach dem Adler-Such-System die Buchstaben finden, als auch für die Zehn-Finger-Tipper. Der Tastenhub ist weder extrem lang, noch besonders gering, ein Buchstabe gilt allerdings schon als getippt, bevor man den Anschlag spürt. Ein sanftes Klicken als Rückkopplung signalisiert, daß der Buchstabe vom Rechner erkannt wurde. Dieses Klicken kommt nicht aus dem Lautsprecher, wie bei einigen anderen Rechnern üblich, sondern ist mechanischen Ursprungs.

Das hat allerdings den Nachteil, daß man den Tastaturklick nicht abschalten kann. Es läßt sich damit aber leben.

Ordnung ist das halbe Leben

Das Gesamtsystem erinnert ein wenig an einen IBM-PC oder einen der vielen Nachbauten. Die Haupteinheit wirkt etwas zierlicher und ist auch leichter als das entsprechende IBM-Produkt. Sie hat zudem noch den Vorteil, daß unter dem Gehäuse der Haupteinheit ein Platz für Tastatur mit Kabel ausgespart wurde.

Ist die Tastatur erst einmal in diesem Fach verschwunden, so benötigt der Amiga angenehm wenig Platz auf dem Schreibtisch und wirkt einfach. Dagegen ist in der Haupteinheit ist kein Platz vorgesehen, um Hardware-Erweiterungen aufzunehmen. Fast alle Erweiterungen müssen außen angeschlossen werden.

Die mechanische Maus des Amiga präsentiert sich in eleganter »Stromlinienform« und liegt gut in der Hand. Sie trägt vorne zwei Knöpfe, die auch beide für die Software benötigt werden. Sie rollt flüssig und verschmutzt deutlich langsamer, als zum Beispiel die Maus des Macintosh.

Das eingebaute Diskettenlaufwerk des Amiga hat (genauso wie das zusätzliche) eine formatierte Kapazität von 880 KByte. Die verwendeten 3½-Zoll-Disketten mit Hartschale sind handlich, robust und zuverlässig. Sie scheinen sich allmählich als Standard bei neueren Mikrocomputern durchzusetzen.

Der Amiga läßt neben einem (oder auch mehreren) zusätzlichen 3½-Zoll-Laufwerken auch 5¼-Zoll-Laufwerke zu. Mit der entsprechenden Software ist er so in der Lage, auch Daten von Disketten zu lesen, die von anderen Computern mit 3½-Zoll oder 5¼-Zoll-Laufwerken stammen.

Der einzige Nachteil, der bei den Diskettenlaufwerken des Amiga vielleicht auffällt, ist ihre Lautstärke. Sie sind verhältnismäßig laut. Durch die Einbauweise erzeugen sie Resonanzschwingungen. Es wäre schön gewesen, wenn sich Commodore/Amiga da etwas mehr Mühe gegeben hätte.

Anders beim RGB-Monitor Amiga 1080. Das ist ein analoger RGB-Monitor, der bei sehr guter Schärfe die volle Farbpalette von 4096 Farben darstellen kann. Neben seinen grafischen Fähigkeiten vermag er aber auch 80spaltigen Text in akzeptabler Qualität zu zeigen. Das Bild ist stabil und fast flimmerfrei, die Farbtrennung bis in die Ecken hervorragend (keine Regenbogen). An die augenschonende Schärfe und Ruhe eines guten Schwarz/Weiß-Bildschirms, wie der des Macintosh oder des Atari ST, kommt er allerdings nicht heran. Dieser Monitor und der Amiga stellen dennoch die wohl ideale Kombination dar. Es gibt aber noch einen zweiten, preiswerten, den 1070. Leider zeichnet er ein etwas verzogenes Bild. Das macht sich besonders in den Ecken bemerkbar.

Wer aber bereits einen IBM-kompatiblen RGB-Monitor, von denen es eine große Auswahl gibt, besitzt, kann sich eine Ausgabe sparen und diesen anschließen. Mehr als 16 Farben sind dann aber nicht möglich.

Wer besonders sparsam sein möchte, kann sich auch zunächst mit einem Farbfernsehgerät behelfen. Das muß vorerst noch ein amerikanischer Fernseher sein, da der Amiga bis jetzt nur einen NTSC-Eingang (amerikanische Farbfernsehnorm) besitzt. Der Farbfernseher liefert aber ein erstaunlich gutes Bild, auf dem 60spaltiger Text noch gut lesbar ist. Bei der endgültigen deutschen Version des Amiga wird der Ausgang der PAL-Norm entsprechen und somit auch zu handelsüblichen Fernsehgeräten passen. Diese endgültige Version soll aber erst ab Sommer 1986 vorliegen. Prototypen der PAL-Version haben wir bereits an einem PAL-Monitor gesehen. Sie lieferten ein wirklich erstaunlich scharfes Bild.

Starke innere Werte

Die Haupteinheit enthält unter anderem die folgenden Komponenten:
- Die CPU M68000
- 256 KByte RAM-Speicher für Programme und deren Daten
- 256 spezieller Kickstart-RAM-Speicher

– drei spezielle Chips für Sound und Grafik
– ein 3½-Zoll-Diskettenlaufwerk mit 880 KByte Kapazität
– diverse Zusatz-Chips für Hilfsaufgaben
– ein Netzteil, das alles mit Strom versorgt.

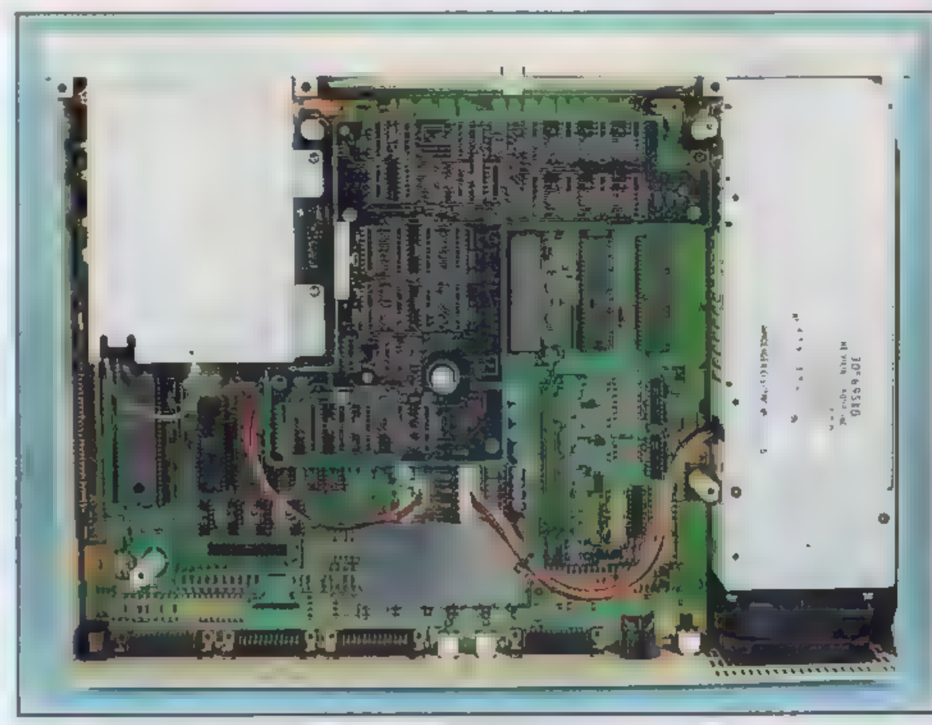
Bei der CPU handelt es sich natürlich um den inzwischen weit verbreiteten Motorola-Prozessor 68000, der auch im Atari ST und Apples Macintosh zu finden ist. Die CPU und der Speicher werden mit einer Taktfrequenz von etwas unter 14,5 MHz versorgt. Die CPU selbst bekommt davon aber nur jeden zweiten Takt ab, so daß sie effektiv nur mit der Hälfte dieser Taktfrequenz (ziemlich exakt 7,19 MHz) läuft. Die anderen Takte sind für die drei Spezial-Chips reserviert.

Mit einem 256 KByte großen RAM-Bereich in der Haupteinheit des Amiga hat es eine spezielle Bewandtnis. Wie fast alle 68000-Computer wurde der Amiga zunächst mit einem großen ROM (Read Only Memory, ein unveränderlicher Speicher, der nur gelesen werden kann) entworfen. Man muß dabei daran denken, daß der Amiga als Heimcomputer gedacht war, der auch ohne Diskettenlaufwerk funktionieren soll. Dieses ROM sollte die wesentlichen Teile des Betriebssystems, der Grafik- und Sound-Software und Hilfsroutinen, die von den späteren Programmen genutzt werden können, enthalten.

RAM statt ROM

Wie scheinbar allmählich üblich, gelang es den Software-Ingenieuren aber nicht, Hardware parallel zur Software fertigzustellen. Man hat deshalb einfach die ROMs aus dem Chassis herausgenommen und zusätzlich RAMs eingebaut. Dieser Speicherbereich hat eine Größe von 256 K und ersetzt das geplante ROM. Beim Einschalten des Amiga muß er von der Kickstart-Diskette mit der vorläufigen Version der Software gefüllt werden, die eigentlich das ROM bereitstellen sollte. Danach kann er nur noch gelesen, aber nicht mehr überschrieben werden (»Write Once Memory« oder kurz »WOM«). Amiga selbst nennt diesen Bereich auch »Writeable Control Store«, »WCS«. (Dieser Name ist in der Computertechnik aber eigentlich schon für andere Dinge reserviert. Wir nennen diesen Speicher deshalb WOM). Da sich der WOM wie ROM verhält, wird in der Dokumentation allerdings oft von ROM gesprochen, wo WOM gemeint ist.

Dieser Ansatz hat Vor- und Nachteile. Obwohl Commodore im Moment verständlicherweise die Vorteile betont. Der Hauptvorteil des Kickstart/WOM-Prinzips liegt in der Flexibilität. Neuere Versionen der ROM-Software können ohne Schwierigkeiten, einfach in Form einer Diskette, an die Kunden ausgeliefert werden. Auch gänzlich andere Betriebssysteme lassen sich realisieren, ohne dafür Teile des stets kostbaren RAMs »verschwenden« zu müssen. Diese Vorteile betreffen aber hauptsächlich den Hersteller und die Softwarehäuser, denen sie die Arbeit erleichtern. Für den Benutzer des Amiga bedeutet der Kickstart-Prozeß einen zusätzlichen Arbeitsvorgang beim Einschalten des Gerätes und natürlich zusätzliche Wartezeit. Nachdem die Kickstart-Software gelesen wurde, muß die Diskette nämlich ausgeworfen und die eigentliche Start-Diskette eingelegt werden. Sie enthält die »äußeren Teile« des Betriebssystems und die Programme, mit denen man arbeiten will.



Amiga-»Striptease«: Aufgeräumt wirkt der Amiga auch von innen

Verbindungen mit der Welt

Der Amiga ist für die Kommunikation mit der Außenwelt gut gerüstet. An seiner Haupteinheit befinden sich reichlich Schnittstellen, die fürs Erste den meisten Anwendern völlig genügen.

Hinten an der Haupteinheit sind unter anderem angebracht:
– ein zusätzlicher Diskettenanschluß
– der Anschluß für die Tastatur
– eine serielle Schnittstelle (nach RS232-Standard)
– eine parallele Schnittstelle (nach Centronics-Standard)
– diverse Anschlüsse für die verschiedenen Monitore
– zwei Lautsprecherausgänge für Stereo-Betrieb

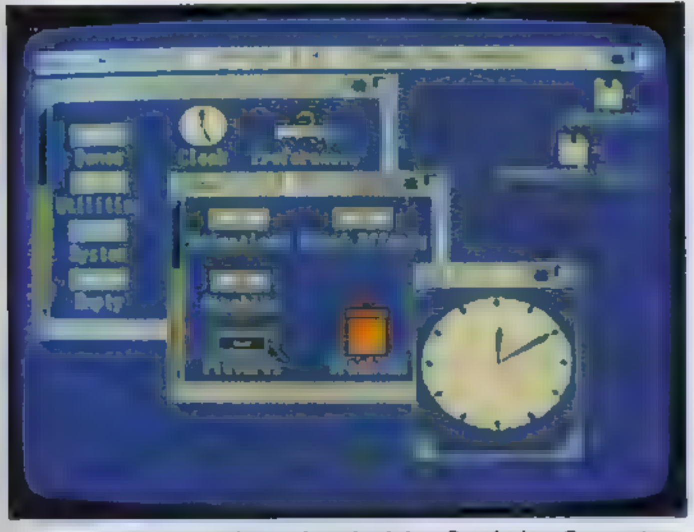
Die wichtigsten dieser Anschlüsse sind die parallele Schnittstelle, die sich für eine Vielzahl handelsüblicher Drucker eignet und die serielle Schnittstelle, an die man ebenfalls Drucker oder ein Modem anschließen kann. Der Monitor-Ausgang und vielleicht auch der zweite Disketten-Anschluß dürften schon im Grundausbau belegt sein.

An der rechten Seite der Amiga-Haupteinheit befinden sich zwei identische Steckbuchsen. Die vordere davon ist stets durch die Maus belegt. Mit der hinteren können Joysticks, Lightpens, Trackballs und ähnliche Peripheriegeräte an den Amiga angeschlossen werden. Vielleicht läßt sich ein findiger Ingenieur aber auch noch einen ganz neuen Verwendungszweck einfallen.

Ebenfalls rechts findet sich, versteckt hinter einer herausnehmbaren Plastik-Leiste, die Erweiterungs-Schnittstelle. Hier werden praktisch alle Signale, die der Amiga intern erzeugt (für Fachleute: der komplette Systembus) nach außen geführt. Hersteller von Zusatzgeräten, aber auch Amiga selbst, bieten bereits eine Reihe von Erweiterungen für diese Steckerleiste an.

Die Vorderseite der Haupteinheit schließlich verbirgt, ebenfalls unter einer Plastikabdeckung, eine Steckerleiste, die speziell für Speichererweiterungen gedacht ist. Amiga selbst bietet eine solche an, die den Speicher von

256 KBytes auf 512 KBytes aufstockt. Mittelfristig ist aber damit zu rechnen, daß andere Hersteller bald auch mit leistungsfähigeren Erweiterungen für diesen Stecker nachziehen. Mindestens 512 KBytes sind unbedingt zu empfehlen. Mit dem Amiga im Grundausbau (256 KBytes) ist nur wenig anzufangen, vor allem, wenn man sich den Dingen widmen will, für die er berühmt ist (Grafik, Multitasking).



Die Amiga Workbench macht mit vielen Symbolen Computerbedienung zum Kinderspiel

Massig Software gratis

Neben der reinen Hardware bekommt man schon beim Kauf des Basispakets eine ganze Menge Software mit. In der amerikanischen Version sind es im Moment die folgenden Disketten (es ist noch nicht ganz klar, welche davon bis zur deutschen Premiere »eingedeutscht« sein werden):

* Die »Kickstart«-Diskette
* Die »Workbench«-Diskette
* Die »Extras«-Diskette
* Die »Polyscope«-Diskette

Die Kickstart-Diskette wird bei jedem Einschalten des Amiga nur einmal benötigt und enthält den »ROM«-Inhalt des Amiga.

Die Workbench-Diskette fungiert als eigentliche »Boot-Diskette« mit der man den Amiga nach dem Start »hochfährt«, wie die Computerleute sagen. Sie sorgt für das Erscheinen der grafischen Benutzerschnittstelle des Amiga, der sogenannten »Workbench«, mit der es der Benutzer wohl die meiste Zeit zu tun haben wird. Sie enthält ansonsten nur einige kleine Utilities und ein paar Demos, mit denen man die Multitasking-Möglichkeiten des Amiga austesten kann.

Auf der Extras-Diskette stellt sich der Amiga durch ein phantastisch gemachtes Tutorial selbst in Bild und Ton vor. Dieses Tutorial macht einen frischgebackenen Amiga-Benutzer mit den wesentlichen Grundzügen der Bedienung seines neuen Prachtstücks vertraut und zeigt zudem noch einige der Fähigkeiten, die den Amiga grafisch und akustisch auszeichnen. Neben diesem Tutorial befindet sich noch das Amiga-Basic auf der Extras-Diskette. Amiga-Basic ist eine sehr komfortable Version des bekannten Microsoft-Basic.

Die Polyscope-Diskette schließlich hält zwei Programme des amerikanischen Softwarehauses »Electronic Arts« für Sie bereit. Das eine Programm zeigt Demos der Produkte, die dieses Softwarehaus im Moment schon an Software für den Amiga zu bieten oder in Planung hat. Das zweite Programm »Polyscope«, eine reine Farborgie, zeigt, was der Amiga in Bezug auf Farbe ohne große Schwierigkeiten zu leisten imstande ist. Wer ein Faible für psychedelische Farbspielereien hat, wird dieses Software-Pröbchen bestimmt begeistert aufnehmen.

Neben diesen drei Disketten sind für Händler noch zwei bis drei Disketten mit zusätzlicher Demo-Software erhältlich. Die meisten werden wohl auch nichts dagegen haben und sie ihren Kunden kopieren. Die Demos machen klar, »was im Amiga steckt«: Viele phantastische Bilder, schnelle bewegte Grafiken, ein einfaches elektronisches Klavier und ähnliches mehr

Zu den Demos gehört natürlich auch der legendäre »Bouncing Ball«, ein rotierender, auf und ab hüpfender, rot/weiß-karierter Globus, der immer, wenn er vom Boden oder vom Rand abprallt, in Stereosound Aufschlaggeräusche von sich gibt.

Dem Einschalten von Haupteinheit und Monitor folgt die bildliche Aufforderung, die »Kickstart«-Diskette einzulegen. Nachdem der Amiga die Kickstart-Diskette geladen hat, verlangt er die »Workbench«-Diskette. Die Workbench ist die Betriebssystem-Oberfläche oder »Shell« des Amiga. Sie entspricht in etwa dem Finder des Macintosh oder dem GEM-Desktop des Atari ST. Nach einigen Copyright-Meldungen und der Aufforderung, die aktuelle Uhrzeit mit Hilfe des Programms »Preferences« zu stellen, erscheint dann auch die Workbench.

Begegnung der anderen Art

Disketten werden als Symbole (Icons) am rechten Bildschirmrand dargestellt. Klickt man zweimal schnell hintereinander auf ein Diskettensymbol, so öffnet sich ein Fenster, in dem der Disketteninhalt dargestellt wird. Dieser besteht aus einzelnen Dateien, die verschiedene Icons besitzen, und Dateiverzeichnisse, die dann wieder weitere Dateien und Dateiverzeichnisse enthalten können. Dateiverzeichnisse werden auf dem Amiga nicht in Form von Ordnern dargestellt, wie bei einigen anderen Rechnern, sondern als eine Art von Schubladen. Jede Schublade kann wie ein Disketten-Icon geöffnet werden und beinhaltet dann wieder Programme, Dokumente und natürlich weitere Schubladen

Die Workbench ist auch die erste Gelegenheit, bei der der Benutzer mit »Intuition« konfrontiert wird. Intuition ist eine Sammlung von Routinen im Amiga-»ROM«, die sich um die Kommunikation der Amiga-Hardware mit dem Anwender kümmert; der Informatiker nennt eine solche Prozedur-Sammlung »Benutzerschnittstelle«. Grafik, der Mauszeiger, Fenster und PullDown-Menüs sind alles Intuition-Konzepte, und nicht Eigenschaften der Workbench. Sie können (und sollten) von allen Programmen auf dem Amiga genutzt werden.

Die Maus des Amiga hat zwei Knöpfe. Um zu den PullDown-Menüs zu gelangen, muß man den rechten Knopf drücken. Der linke dient zum Anklicken (Auswählen) und Verschieben von Objekten auf dem Bildschirm.

Sobald der rechte Knopf betätigt ist, erscheint am oberen Rand des Bildschirms eine Zeile mit den Titeln der verschiedenen Menüs. Bewegt man den Maus-Zeiger über einen solchen Titel, klappt darunter das Menü hervor, aus dem man dann einen Punkt auswählen kann. Sobald sich die Maus über einem Menü-Punkt befindet, wird

dieser hervorgehoben. Das kann durch eine Farbänderung zustande kommen oder durch ein Umrahmen des gewählten Punktes mit einer dünnen Linie. Läßt man den rechten Mausknopf los, während ein Menü-Punkt hervorgehoben wird (oder drückt man kurz den linken Knopf), so gilt dieser Befehl als erteilt.

Amiga-Menüs können Untermenüs enthalten. Kommt man zu einem Menü-Punkt mit zusätzlichem Untermenü, so klappt rechts neben dem Hauptmenü ein (meist kleines) Untermenü heraus. Damit verfährt man ebenso wie mit dem Hauptmenü. Untermenüs stellen sich oft als sehr praktische Sache heraus. Sie sind andererseits aber auch etwas schwierig zu bedienen, und es verlangt einige Geschicklichkeit beim Umgang mit der Maus. Etwas gewöhnungsbedürftig für diejenigen, die schon andere Fenstersysteme kennen, sind die Fenster des Amiga. Sie können, wie bei »ordentlichen Fenstern« zu erwarten, bewegt werden, indem man sie am oberen Rand ergreift. Zum Verkleinern oder Vergrößern verschiebt man an der rechten unteren Ecke den »Größen-Gadget«. Sie können geschlossen werden, indem man einen Knopf in der oberen linken Ecke (das »Schließ-Gadget«) drückt. »Gadgets« sind die Schalter und Instrumente eines Cockpits, das man mit Intuition gebaut hat. Sie zeigen Werte in grafischer Form an und reagieren auf Mausklicks meist mit einem bestimmten Verhalten.

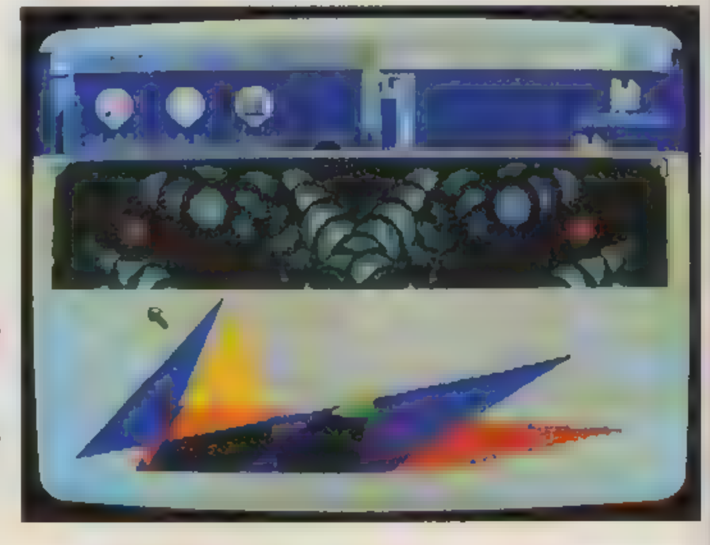
Blick durchs Fenster

Amiga-Fenster können sich auch überlappen, wodurch ein kleines Ärgernis beginnt. Will man ein Fenster nach vorne holen, weil es vielleicht ein anderes Fenster teilweise verdeckt, muß man einen ganz bestimmten »Knopf« (das »Front-Gadget«) in der oberen, rechten Ecke des Fensters anklicken. Besonders praktisch ist das, wenn genau dieser Knopf verdeckt wird! Dann heißt es, in die Hände spucken, und die anderen Fenster erst mal »beiseite räumen«. Dafür gibt es aber einen zweiten Knopf (das »Back-Gadget«), mit dem man ein Fenster »ganz nach hinten« legen kann, wenn man an diesen Knopf dran kommt. Mit dieser sinnreichen Konstruktion hat Commodore-Amiga es den Benutzern unnötig schwer gemacht.

Eine weitere etwas lästige Eigenschaft der Intuition-Fenster ist, daß sie immer komplett auf dem Bildschirm zu sehen sein müssen abgesehen von eventuellen Verdeckungen durch andere Fenster. Man kann ein Fenster nicht so weit »beiseite legen«, bis nur noch ein kleiner Teil davon zu sehen ist. Zum »beiseite legen« muß man das Fenster vorher erst auf Minimalformat verkleinern und später, wenn man das Fenster wieder braucht, auf die alten Ausmaße vergrößern. Eine lästige und überflüssige Einschränkung.

Intuition-Fenster können natürlich auch gescrollt werden. Das heißt, der Ausschnitt eines eventuell größeren Bildes, der im Fenster-Inneren dargestellt wird, läßt sich verschieben. Dazu dienen, wie auch bei Mac und beim Atari-ST, Rollbalken. Ein weißer Kasten in der Mitte des Rollbalkens symbolisiert den dargestellten Ausschnitt im Verhältnis zur Gesamtheit des Bildes. Mit den Pfeilen an beiden Enden der Rollbalken kann dieser Ausschnitt in kleinen Schritten (bis hinunter zu einem Pixel) bewegt werden. Man kann aber auch den weißen Kasten in der Mitte des Rollbalkens ergreifen und ihn direkt an der beabsichtigten Stelle plazieren.

Eine seiner hervorragenden Eigenschaften ist die Fähigkeit, mehrere Programme, »Tasks« oder »Prozesse« gleichzeitig laufen zu lassen.



Ähnlich wie Fenster können auch die visuellen Bildschirme, die Screens, übereinanderliegen. Hier überlappen sich drei Bildschirme.

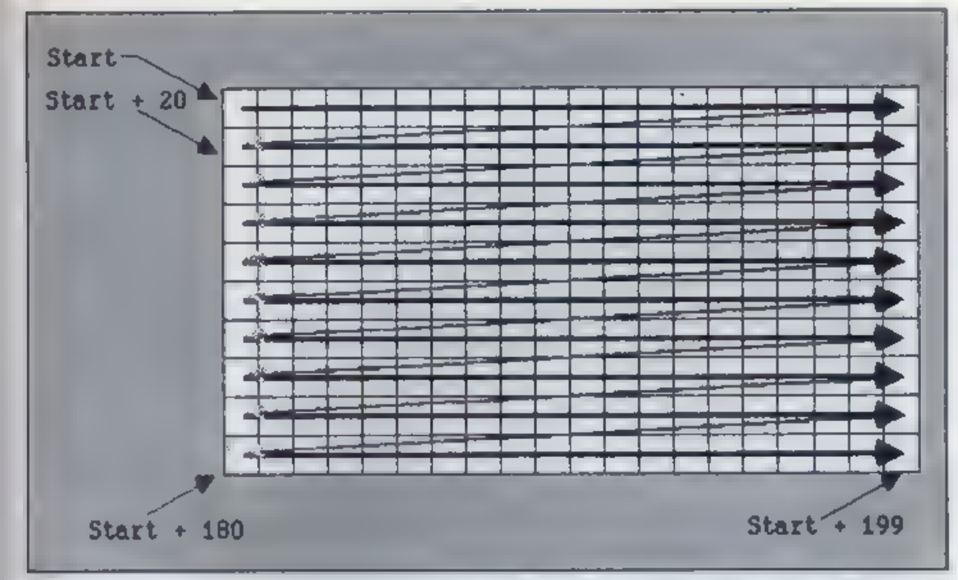
Arbeiten für drei

Diese Eigenschaft, das »Multitasking«, ist nicht zuletzt deshalb erwähnenswert, weil sie bisher nur auf wesentlich größeren und teueren Computern üblich war.

Startet man zum Beispiel von der Workbench ein Programm, »übernimmt« es deshalb normalerweise nicht den ganzen Computer. Es öffnet ein neues Fenster und erledigt dort seine Ein- und Ausgaben. Falls es mehrere Fenster braucht, so kann es diese nacheinander öffnen oder sogar einen virtuellen Bildschirm anlegen, der einen eigenen Speicher und einen anderen Grafikmodus besitzt. Die anderen Programme aber, die zu diesem Zeitpunkt schon liefen, arbeiten einfach weiter. Das hört sich im ersten Moment ganz toll an. Andererseits fragt man sich dann bald auch, wozu das Ganze? Schließlich kann man ja keine zwei oder drei Programme gleichzeitig bedienen, selbst wenn sie gleichzeitig laufen können. Dieser Einwand stimmt, und man sollte die Vorteile des Multitaskings nicht überschätzen. Insbesondere auf dem Amiga mit nur 256 KByte Hauptspeicher läuft in Sachen Multitasking sowieso nur wenig. Jedes Programm braucht natürlich Platz im Hauptspeicher und davon ist recht rasch nicht mehr genügend da. Selbst die recht simplen Demos, auf der Workbench-Diskette mitgeliefert, schlucken hungrig Speicher. Wer also das Multitasking des Amiga nutzen will, dem ist die 512K-Version zu empfehlen.

Andererseits entpuppt sich Multitasking gerade bei langwierigen Vorgängen aber auch als praktische Sache. Die meisten Druckvorgänge laufen auf dem Amiga standardmäßig im Hintergrund, also als separater Task. Während der Drucker rattert, kann man sich anderen Dingen zuwenden. Genauso kann während des Formatierens einer Diskette bereits mit anderen Programmen weitergearbeitet werden. Für integrierte Programmpakete eröffnen sich ebenfalls völlig neue Möglichkeiten.

Im Moment fehlen von seiten der Workbench noch Möglichkeiten, die verschiedenen Tasks zu beeinflussen. So kann man zum Beispiel einen Task nicht in einen »Schlafzustand« versetzen, damit er vorübergehend die anderen Tasks nicht bremst. Jedes Programm, das zusätzlich gestartet wird, verlangsamt deshalb alle anderen Programme merklich. Maximal drei intensiv rechnende Programme, die gleichzeitig arbeiten, sind das höchste der Gefühle.



Die Zuordnung von Bitadressen zu den jeweiligen Bildschimpositionen im Speicher

Grafikauflösung	Speicherbedarf (pro Bitebene)
320 (H) x 200 (V)	8 KByte
640 (H) x 200 (V)	16 KByte
320 (H) x 400 (V) (interlaced)	32 KByte
640 (H) x 400 (V) (interlaced)	64 KByte
320 (H) x 200 (V) (H.A M)	56 KByte
320 (H) x 400 (V) (H.A.M.,Interl.)	112 KByte

Tabelle: Grafik-Modi des Amiga

Die Trägheit bei Programmeingaben oder auch nur beim Bewegen eines Fensters mit der Maus übersteigt dann jegliche Akzeptanz.

Starken Anteil am Multitasking haben die Spezial-Chips des Amiga. Sie beschleunigen zum einen den Wechsel von einem Programm zum anderen, zweitens den Ablauf des Programmes selbst und zum dritten beschleunigen sie die Programmabarbeitung, da sie zum Beispiel Grafik wesentlich schneller erzeugen als der 68000 Prozessor.

Gummibildschirm im Einsatz

Da jederzeit verschiedene Programme, die teils unterschiedliche Anforderungen an die Bildschirmauflösung oder die Anzahl möglicher Farben stellen, gleichzeitig aktiv sein können, sieht Intuition »virtuelle Bildschirme«, sogenannte »Screens«, vor. Sie haben dieselbe Größe wie der »wirklich« physikalische Bildschirm, sind aber wesentlicher flexibler. Benötigt ein Programm bestimmte Bildschirmeigenschaften, zum Beispiel 320 x 200 Bildpunkte in 32 Farben, so eröffnet es beim Start einfach einen eigenen Bildschirm und auf diesem dann seine Fenster. Genügen die Standardeigenschaften des Workbench-Bildschirms, so arbeitet es auf diesem. Jeder dieser Bildschirme darf eine andere Grafikauflösung bieten. Begrenzt ist das durch den zur Verfügung stehenden Speicherplatz.

Ähnlich wie Fenster können auf dem physikalischen Bildschirm mehrere virtuelle Bildschirme übereinander liegen und sich teilweise verdecken. Gegeneinander verschiebbar kann sich auch ihre Reihenfolge mit Back- und Front-Gadgets (in der Menü-Zeile) oder über bestimmte Tastaturcodes ändern. Im Gegensatz zu Fenstern haben Screens aber immer die volle Bildschirmbreite, was ein seitliches Verschieben ausschließt. Das geht nur bei den Playfields. Es ist sehr eindrucksvoll, wenn man einen virtuellen Bildschirm »hinter« den Workbench-Bildschirm schiebt, während sich darin Grafik-Operationen abspielen. Virtuelle Bildschirme machen ein echtes Multitasking auf einem so vielseitigen Grafikcomputer überhaupt erst möglich.

Neben der grafischen Benutzerschnittstelle bietet der Amiga für diejenigen, die von der Tastatur nicht lassen können, auch eine Schnittstelle à la MS-DOS. Auf dem Amiga heißt diese Einrichtung »Command Line Interface« oder kurz CLI.

Das CLI liegt ein wenig versteckt, und es verlangt schon einige Einsicht in das System, um es überhaupt zu aktivieren. Hinzu kommt, daß sein Vorhandensein das Benutzerhandbuch nur einmal andeutungsweise erwähnt und die entsprechenden Befehle natürlich auch nicht dokumentiert.

Hat man aber endlich ein CLI-Fenster geöffnet – oder auch mehrere, der Amiga ist ja Multitasking-fähig – ist eigentlich alles, was die Workbench mit Maus und Menüs erledigt, auch mit Tastaturbefehlen zu erreichen. Den Inhalt einer Diskette, genauer eines Datenverzeichnisses, kann man mit dem Befehl »DIR« in Textform auflisten. Wer Icons nicht mag, wird sofort begeistert sein!

Verborgene Schätze

Vom CLI aus gelangt man auch an die verborgenen Schätze der Workbench-Diskette. Unter anderem befindet sich darauf nämlich, von der Workbench aus völlig unsichtbar, ein kompletter Fullscreen-Editor, ein zeilenorientierter Editor und diverse Utilities. Für alle diese Programme kann man die Maus und Grafikfähigkeiten des Amiga vergessen, dafür ihre Ein-/Ausgabe-Dateien aber umleiten und ihnen bereits beim Start Parameter mitgeben, die ihnen sagen, was sie tun sollen. Natürlich ist es auch möglich, eine Reihe von Befehlen zu Kommandofolgen zusammenzufassen. Die dafür verwendete Kommandosprache kennt sogar IF-THEN-ELSE-Konstrukte, Marken, sowie Variablen und Parameter.

Alles das wirkt ein wenig antiquiert im Vergleich mit einer modernen Benutzerschnittstelle. Man sollte aber berücksichtigen, daß solche hochentwickelten Benutzerschnittstellen wie »Intuition«, die Ressourcen eines Computers recht stark beanspruchen und einige Aufgaben bis heute nicht so gut bewältigt, wie die primitiveren, die für ein simples Terminal entwickelt wurden.

RAM-Disk spart Platz

Als zusätzliches Bonbon gelangt man vom CLI aus auch noch an eine RAM-Disk. (Das ist ein im RAM simuliertes Diskettenlaufwerk.) Diese RAM-Disk ergibt bei einem Speicherausbau von 256K zwar nicht viel Sinn, ist aber eine nützliche Einrichtung für die Zukunft. Die RAM-Disk des Amiga ist besonders effizient. Anders als bei den anderen Computern muß hier nicht ein bestimmter Speicherbereich reserviert werden. Die Dateien nehmen im Speicher immer nur soviel Platz ein, wie sie benötigen. Wird eine neue Datei in die RAM-Disk kopiert, so »zwackt« sie sich ein Stück

von genau der Größe der Datei im Speicher ab. Dieses wird sofort wieder freigegeben und kann als ganz normaler RAM-Speicher weiter verwendet werden, sobald die Datei wieder gelöscht ist.

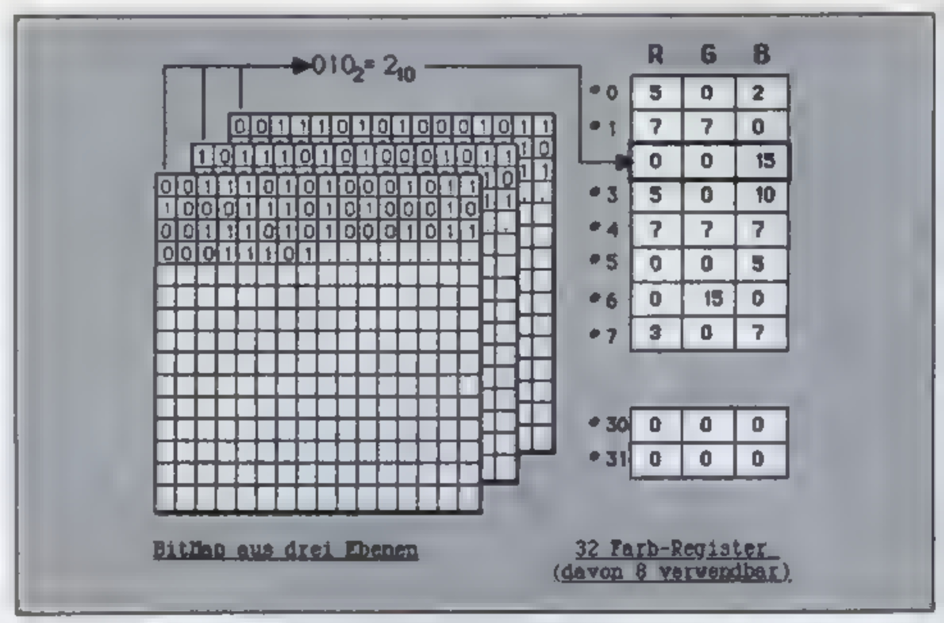
Der mitgelieferte Texteditor »ed« läßt zwar traurigerweise die Maus ziemlich links liegen, bietet sonst aber in etwa alle Möglichkeiten, die die Bearbeitung einfacher Texte, zum Beispiel Programm-Sources, erfordert. Neben dem Fullscreen-Editor »ed« gibt es auch noch einen zeilenorientierten Texteditor namens »edit« in guter alter »edlin«-Tradition. Er ist umständlich zu bedienen, bietet gegenüber »ed« aber andere Vorteile, die in diesem knappen Platz aber kaum erörtert werden können.

Diejenigen, die schon einmal Bekanntschaft mit dem Editor EMACS gemacht haben, wird es freuen zu hören, daß eine vereinfachte Version von EMACS, Micro-EMACS, auch bereits für den Amiga zu haben ist. Sie ist Public Domain und wird, sobald noch weitere PD-Software für den Amiga aufgetaucht ist, im Rahmen des Software-Services für einen geringen Unkostenbeitrag zu erhalten sein. Micro-EMACS nutzt sogar die Fenster und (neben den üblichen »CTRL«- und »ESC«-Codes) die Menüs des Amiga und erlaubt die gleichzeitige Bearbeitung zweier Texte in einem geteilten Fenster.

Punkt, Punkt, Komma, Strich

Ein Aspekt, der am Amiga nahezu uneingeschränkt gelobt wird, sind seine Grafikfähigkeiten. Sie übertreffen diejenigen vergleichbarer Computer in vieler Hinsicht weit. Drei speziell für diesen Zweck entworfene und für Commodore/Amiga gebaute Chips, sogenannte »Custom VLSI-Chips« sorgen für eine Vielfalt von Leistungsmerkmalen. Vor allem für eine Geschwindigkeit, wie man sie bisher – zumindest bei Computern in dieser Preislage – für unmöglich hielt.

Ein komplettes Bild, das wir im Monitor des Amiga sehen, besteht im allgemeinen immer aus drei Komponenten. Zwei davon sind im wesentlichen statisch und die dritte dient vor allem der Animation. Am Rand des Bildschirms und an allen Stellen, an denen keine Grafik gezeichnet wurde, ist der Bildschirmhintergrund zu sehen. Er hat eine gleichmäßige Farbe, die mit den Preferences beliebig geändert werden kann. »Darüber« liegt die zweite statische Komponente, das »Playfield« oder zu deutsch »Spielfeld«, das aus den



Die Zuordnung von Farben zu den einzelnen Bildschirmpunkten

unveränderlichen oder sich nur langsam ändernden Grafiken besteht. »Vor« oder »über« dem Playfield schließlich stehen noch die Sprites, kleine Bilder, die sehr einfach und schnell über der restlichen Grafik hin- und herbewegt werden können. Zunächst aber zum Playfield und dann zur animierten Grafik.

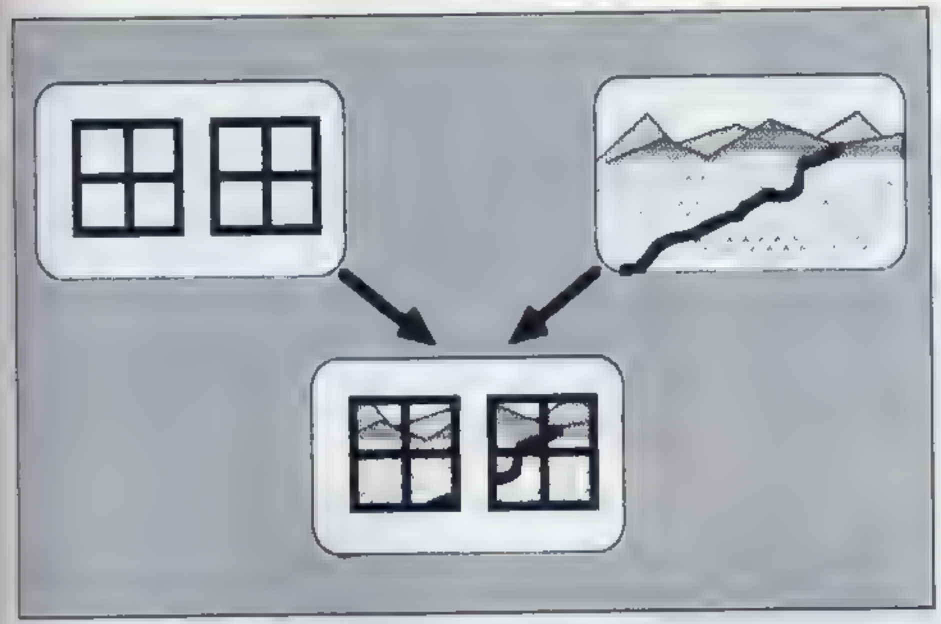
Die Grafik des Amiga ist »bitmapped«. Dies bedeutet, daß die Farbe jedes Punktes auf dem Bildschirm von einem oder mehreren Bits im Speicher abhängt. Den simpelsten Fall ergibt eine 1:1-Zuordnung, bei der jeder Punkt auf dem Bildschirm genau einem Bit im Speicher entspricht. Da der Amiga mit mehreren Farben gleichzeitig in einer Grafik spielt, liegt der Fall natürlich etwas komplizierter. Zur Veranschaulichung der wichtigsten Grundlagen gehen wir zunächst aber von einer reinen Schwarzweiß-Grafik aus.

Grundlage: Schwarzweiß

Während der Elektronenstrahl über die Bildröhre fährt, holt der Videochip nacheinander die entsprechenden Bits aus dem Speicher und modifiziert den Strahl entsprechend. Ist das Bit gleich 1, so läßt er den Strahl in voller Stärke auf die Phosphor-Schicht der Bildröhre prallen. Ein heller Punkt entsteht. Ist das Bit gleich 0, so drosselt er den Strahl, so daß der Punkt schwarz bleibt. Die Abfolge, in der der Elektronenstrahl über den Bildschirm fährt, entspricht im simpelsten Fall genau der Reihenfolge der Bits im Speicher. Während der Strahl aber von links nach rechts und von oben nach unten schwenkt und dabei einen rechteckigen Bereich abdeckt, liegen die Bits im Speicher natürlich alle hintereinander bei steigenden Adressen.

Man kann sagen, daß bei einer Bitmapped Grafik eine eindeutige Zuordnung von einem Bit im Speicher (bei Schwarzweiß- oder genauer zweifarbiger Grafik) zu einem Punkt am Bildschirm besteht. Die Bits bilden eine »map«, die der Videologik zeigen, an welchen Stellen der Bildschirm welche Farbe haben soll.

Möchte man mehr als zwei Farben in einer Grafik verwenden, reicht ein Bit nicht mehr aus, um alle nötigen Farbinformationen eines Bildschirmpunktes zu liefern. Ein Bit kann nur zwei Zustände haben (0 und 1), denen genau zwei Farben zugeordnet werden können. Es gilt deshalb: N Bits reichen zur Codierung von 2^N Farben. Jeder dieser Zahlen wird nun eine Farbe zugeordnet und die Videologik eines Computers kann nun nicht nur ein Bit auf einen Bildschirmpunkt in einer von zwei Farben (zum Beispiel schwarz und weiß) abbilden, sondern drei Bits auf einen Punkt, der insgesamt acht verschiedene Farben haben kann. Dieses Prinzip findet heute bei vielen Computern Anwendung. Unterschiede gibt es nur dabei, wie die acht Zahlen (bei drei Bits) bestimmten Farben zugeordnet werden. Bei einigen Modellen macht man sich die Tatsache zunutze, daß Farbfernsehbilder aus den drei Grundfarben rot, grün und blau durch additive Farbmischung zusammengesetzt werden. Jedes der drei Bits, die durch die Videologik auf einen Bildschirm abgebildet werden, sind deshalb einer der drei Grundfarben zugeordnet. Jede Farbe, deren Bit gleich 1 ist, wird dann dem jeweiligen Punkt »beigemischt«. Ist das Bit gleich 0, entfällt die Farbe. Die


Zwei Playfields wurden zu einem Bild überlagert

Farben, die sich auf diesem Wege ergeben, entsprechen dabei den acht möglichen Kombinationen von drei Bits.

Eine wesentlich flexiblere Methode ist es, die acht möglichen Werte der drei Bits über »Umwege« bestimmten Farben zuzuordnen. Beim Amiga bedeutet jede dieser Zahlen von 0 bis 7 die Nummer eines »Farbregisters«. Erst in diesem Farbregister steht die tatsächliche Farbe, die am Bildschirm erscheint. Der Amiga hat allerdings nicht nur acht Farbregister, sondern 32, und kann somit maximal 32 verschiedene Farben gleichzeitig auf dem Bildschirm produzieren (In Wirklichkeit noch ein paar mehr; dafür werden aber noch andere Tricks benötigt).

Beim Zeichnen des Bildes zu einer bestimmten Bitmap im Speicher hat es die Videologik nun allerdings etwas schwerer. Sie schaut zunächst nach der Zahl in der Bitmap und verwendet diese, um ein Farbregister anzuwählen. Dessen Inhalt, der natürlich auch ein bestimmtes Format hat, bestimmt dann die endgültige Farbmischung für den Bildschirmpunkt.

Farbenspiel

Die Farbregister des Amiga sind 12 Bits breit. 12 Bits bedeuten 4096 verschiedene mögliche Kombinationen. Jeweils 4 Bits davon sind einer der drei Grundfarben zugeordnet. Jede Grundfarbe kann also insgesamt 16 ($=2^4$) verschiedene Intensitäten oder Anteile zu einer Farbe beitragen. Statt jede dieser drei Farben nur an- oder ausschalten zu können, wie bei der einfachen Lösung ohne Farbregister, kann man sie »fein dosiert« anwenden

Anders als bei manchen anderen Computern liegen im Amiga die drei Bits, die bei acht Farben die Farbe eines Grafikpunktes bestimmen, im Speicher nicht nebeneinander. Die Bits für einen Punkt befinden sich in drei verschiedenen Ebenen, das heißt Speicherbereichen. In jeder Ebene liegen die Bits für aufeinanderfolgende Punkte auch hintereinander. Dies hat entscheidende Folgen für die Grafik-Software. Das kleine Programm, das einen Kreis zeichnet, indem es die entsprechenden Bits in einer Bitmap setzt, kann genauso für zwei, drei oder sechs Bitmaps verwendet werden. Es muß nur zwei-, drei- oder sechsmal hintereinander ausgeführt werden – für jede Farbebene einmal.

Da der Amiga maximal 32 Farbregister nutzen kann, erscheinen also mit der hier vorgestellten Methode der Bitmapped Grafik mit Farbregistern maximal 32 der 4096 möglichen Farben gleichzeitig am Bildschirm. Für diese 32 Farben werden 5 Bit-Ebenen benötigt ($2^5 = 32$). Es gibt aber auch einen Modus, der sechs Bit-Ebenen benötigt und theoretisch alle 4096 möglichen Farben gleichzeitig am Bildschirm zeigt. Dieser Modus heißt Hold-And-Modify-Modus oder kurz HAM. Der Name bedeutet soviel wie »Festhalten und Modifizieren«.

Die sechs Bits, die im HAM-Modus einem Bildschirm-Punkt zugeordnet sind, werden nicht, wie in allen anderen Modi, als Nummer eines Farbregisters interpretiert, sondern enthalten manchmal direkt Farbinformationen. Sie werden von der Video-Hardware wie die Instruktionen eines Computers aufgelöst.

Sind die ersten beiden Bits für einen Punkt beide gleich Null, so werden die restlichen vier Bits als die Nummer eines Farbregisters interpretiert, das die Farbe dieses Punktes enthält. In diesem Fall wirkt der HAM-Modus als ein etwas speicherverschwendender Grafikmodus mit vier Bit-Ebenen (=16 Farben).

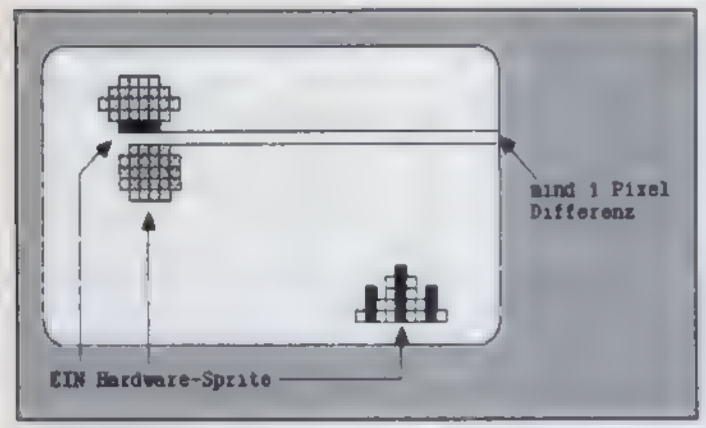
Sind die ersten beiden Bits für einen Punkt aber gleich 01, so hält die Video-Hardware die Farbe des vorangehenden Bildschirmpunktes fest und ändert nur dessen Rot-Komponente. Die restlichen vier Bits des Punktes enthalten den neuen Rot-Wert für diesen Punkt, die Blau- und Grün-Werte bleiben erhalten. Die so aus dem Nachbarpunkt und einem neuen Rot-Wert zustande kommende Farbe ergibt die Farbe des neuen Punktes.

Sind die ersten beiden Bits für einen Punkt aber gleich 10, so wird nur der Grün-Wert geändert; sind die Bits gleich 11, so enthalten die restlichen vier Bits den Blau-Wert und der Rot- und Grün-Wert wird beibehalten. In allen drei Fällen ist es übrigens egal, wie die Farbe des vorangegangenen Punktes, die durch einen neuen Wert für eine der drei Farb-Komponenten modifiziert wird, zustande kam. Sie kann entweder direkt aus einem Farbregister stammen oder auch schon modifiziert gewesen sein.

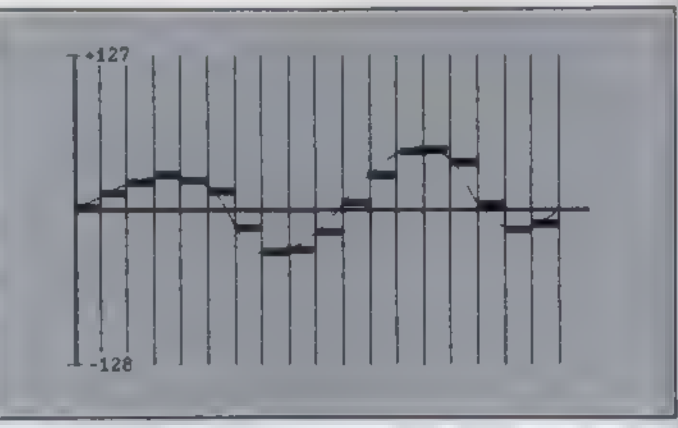
Einfach nur bunt

Wie leicht zu sehen ist, lassen sich auf diese Weise alle 4096 Farben erzeugen. Es ist zwar nicht möglich, jeden beliebigen Punkt auf jede beliebige Farbe zu setzen, aber mit etwas Geschick kann man diese Einschränkung recht gut umgehen. Nach spätestens drei Punkten gelangt man immer zu der beabsichtigten Farbe. Wieviele Bits für einen Punkt »zuständig« sind und damit, wieviele Farben zur Verfügung stehen, wie hoch die Auflösung des Bildschirms ist, und wie die Bits des Punktes interpretiert werden, kann der Programmierer nach den Bedürfnissen seines Programms frei bestimmen. Die Tabelle enthält eine Zusammenfassung der möglichen Grafikmodi.

Obwohl das Playfield grundsätzlich eine statische Angelegenheit ist, kann es bewegt werden. Falls ein Bild größer ist, als der gewählte Grafik-Modus zuläßt, kann man den Bildschirm als »Fenster« verwenden. Nicht zu verwechseln mit den Intuition-Fenstern, die auf einer völlig anderen Software-Ebene arbeiten. Die komplette BitMap für das ganze Bild muß dazu aber im Speicher des Amiga bereitstehen und kostet viel Platz. Einfach, indem man ein oder zwei Register in der Video-Hardware ändert, kann man dann den Ausschnitt des großen Bildes manipulieren. Ein solches Schwenken des



Sprites können weiterverwendet werden



Ein Klang wird digitalisiert

Fensters, auch Scrolling genannt, ist sowohl in horizontaler als auch in vertikaler Richtung problemlos möglich.

Und schließlich serviert der Amiga bei Bedarf auch zwei Playfields gleichzeitig am Bildschirm. Man kann zwei komplette Bildschirme übereinander legen. Das »obere« Playfield läßt an allen Stellen, an denen es die Farbe »Transparent« (Farbregister 0) hat, das dahinterliegende Playfield durchscheinen. Dies ist besonders dann praktisch, wenn eine Grafik auch konzeptionell zwei verschiedene Schichten hat und eine davon vielleicht noch bewegt werden soll.

Beide Playfields können verschiedene Farben haben, müssen aber dieselbe Bildschirm-Auflösung verwenden. Das Scrolling der beiden geschieht unabhängig voneinander. Die Verwendung von zwei Playfields hat nur – wie immer – einen entscheidenden Nachteil: sie kostet doppelt soviel Platz wie ein einzelnes Playfield. Berücksichtigt man, daß selbst ein relativ bescheidenes Playfield, wie das der Workbench, 32 KByte benötigt, ist das ein nicht geringer Minuspunkt.

Die einfachste animierte Grafik auf dem Amiga sind Sprites. Der Zeiger, der den Mausbewegungen folgt, ist zum Beispiel ein Sprite. Der Amiga unterstützt 8 Sprites durch Hardware; durch Tricks können allerdings wesentlich mehr eingesetzt werden. Jedes Sprite ist eine kleine BitMap, deren Inhalt von der Video-Hardware dem Playfield bei der Anzeige überlagert wird. Die Playfield-BitMap wird dabei nicht modifiziert. Die Position, an der ein Sprite auf dem Bildschirm über dem Playfield auftaucht, bestimmen für jedes Sprite zwei Register (für die X- und Y-Position) in der Video-Hardware. Einfach, indem der Inhalt dieser Register geändert wird, können die entsprechenden Sprites auf dem Bildschirm bewegt werden. Dies geht natürlich sehr schnell!

Dafür unterliegen Sprites aber auch einigen Einschränkungen. Die erste ist natürlich die Anzahl der Sprites. Gerade für Videospiele werden acht bewegte Objekte nicht genug sein. Aber die Designer des Amiga haben sich eine Lösung für dieses Problem einfallen lassen: Die Hardware, die ein Sprite auf dem Bildschirm darstellt, wird ja nicht mehr benötigt, sobald das »untere Ende« dieses meist kleinen Bildes erreicht ist. Erst wenn der Elektronenstrahl am unteren Ende des Bildschirms angelangt und wieder nach oben zurückgekehrt ist, wird diese Hardware wieder benötigt. Warum sollte man sie ab diesem Moment, wo sie mit dem einen Sprite erst einmal »fertig ist«, nicht für andere Dinge verwenden?

Genau das ist möglich! Ist das untere Ende eines Sprites auf dem Bildschirm erreicht, kann durch dieselbe Sprite-Hardware ein neues Sprite an einer anderen Position gezeichnet werden. Die beiden dürfen sich – logischerweise – nicht überschneiden. Ist dieses zweite Sprite fertig, kann ein Drittes folgen. Auf diese Weise können die acht Hardware-Sprites nahezu beliebig oft »wiederverwendet« werden – solange die »Folge-Sprites« alle untereinander liegen und von ihrem Vorgänger mindestens um ein Bildschirmpixel getrennt sind.

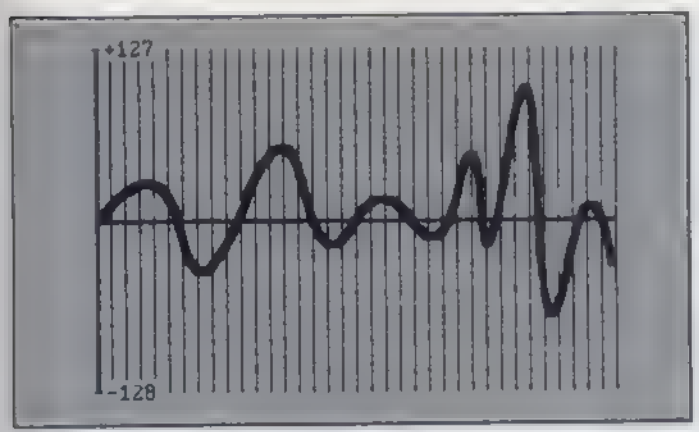
Bahn frei für BOB!

Eine andere »unangenehme« Eigenschaft von Sprites ist ihre Breite. Sprites können zwar beliebig hoch, aber nur ein Speicherwort (16 Bits) breit sein. Diese Einschränkung ist wegen der Hardware nicht zu umgehen. Man kann zwar zwei Sprites (oder auch mehr) zu einem zusammenfassen, dabei wird die Zahl der Sprites, die einem zur freien Verfügung stehen, immer kleiner. Überdies beschränkt sich die Farbenanzahl der Sprites auf vier verschiedene. Eine davon ist transparent, weshalb nur drei echte Farben übrig bleiben. Diese kommen zudem aus derselben Farbpalette, die auch für die Farben des Playfields verwendet wird. Die Farben von Sprites und Playfields können also nicht unabhängig voneinander verändert werden. Falls das Playfield aber nicht alle 32 Farben nutzt und nicht alle Sprites verwendet, wie es in der Workbench der Fall ist, stört diese Abhängigkeit wenig.

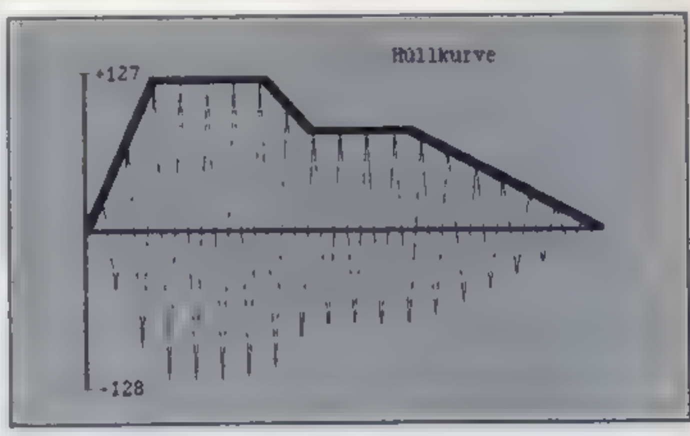
Alle diese Einschränkungen gibt es bei BOBs nicht. BOBs, Blitter Objekts, sind ebenfalls kleine BitMaps, wie auch die Sprites. Sie unterliegen aber keiner Beschränkung in der Breite und können soviele Farben haben, wie auch das Playfield. Sie lassen sich aber bei weitem nicht so schnell bewegen wie Sprites und beanspruchen den Computer (sowohl den Prozessor, wie auch den Speicher) wesentlich stärker.

Bei BOBs wird nämlich die kleine BitMap, die ihr Aussehen enthält, wirklich in die Playfield-Bitmap hineinkopiert. Vorher sichert das Betriebssystem aber den alten Inhalt der Playfield-Bitmap an dieser Stelle in einem Puffer. Beim Bewegen eines BOBs müssen also wirklich Daten im Speicher verschoben werden. Je größer das BOB, desto mehr Daten sind es. Das kostet wesentlich mehr Zeit, als das Laden eines Registers mit einem neuen Wert. Gäbe es nicht den Blitter, der speziell für solche »Datenschaufeleien« geschaffen ist, wären BOBs viel zu langsam, um überhaupt praktikabel zu sein.

Die nächsthöhere Stufe bei der Erzeugung bewegter Grafiken im Amiga sind die »AnimObjects«. AnimObjekts, eine Reihe von BOBs, sind in einer besonderen Datenstruktur zusammengefaßt. Sie enthalten unter anderem mehrere verschiedene BOBs, die gegeneinander ausgetauscht werden. Dadurch entsteht der Eindruck, als bewegten sich diese Objekte. Die Reihenfolge und Geschwindigkeit, in der die verschiedenen BOBs gegeneinander ausgewechselt werden, hängt rein vom Programmierer ab. Gleichzeitig kann er dem AnimObjekt noch die Geschwindigkeit mitteilen, mit der es sich über den Bildschirm bewegen soll und eine Beschleunigung, die angibt, mit welcher Rate sich diese Geschwin-



Ein digitalisierter Klang (Ausgabe)



Hüllkurve eines Tones

digkeit ändert. Die Software, die die AnimObjekts bewegt, registriert dabei sogar, wenn zwei solche Objekte zusammentreffen. Ein solcher Zusammenstoß kann sogar eine bestimmte Prozedur eines Programms auslösen.

Humb, Tata

Nicht ganz so große Publizität wie die Grafikfähigkeiten, haben die Sound-Möglichkeiten des Amiga gefunden. Sie sind nichtsdestotrotz recht beeindruckend. Der Amiga besitzt insgesamt vier unabhängige Tonkanäle, von denen jeweils zwei zu einem Stereo-Kanal zusammengefaßt werden können. Jeder der vier Kanäle kann eine völlig beliebige Wellenform mit beliebiger »Hüllkurve« ausgeben. Es erfordert dazu nur in sehr geringem Maße ein Eingreifen des Programms, das den Ton irgendwann einmal erzeugt hat.

Man darf von der Sound-Hardware des Amiga natürlich nicht zuviel verlangen. Es handelt sich schließlich nicht um einen Synthesizer, und die Klangqualität kann Hi-Fi-Tests gewiß nicht standhalten. Für die Darstellung eines Klangs werden im Amiga zum Beispiel 8 Bit breite Werte verwendet. Hi-Fi-Geräte in Digital-Technik verwenden üblicherweise 16 Bit, was zu einer rund 250mal besseren Klangqualität führt – von anderen Gesichtspunkten ganz abgesehen.

Um zu verstehen, wieso der Amiga trotzdem relativ natürliche Klänge erzeugen kann, betrachten wir einmal, wie die Amiga-Sounds zustande kommen. Ein Klang kann entweder völlig frei erfunden oder aus einem natürlichen Klang digitalisiert werden. Dazu mißt man einfach die Lautstärke eines natürlichen Klangs in sehr kurzen Zeitabständen. Sind diese Zeitabstände kurz genug, so mißt man dabei nicht mehr den Verlauf dessen, was wir normalerweise »Lautstärke« nennen würden, sondern den Klangverlauf. Diesen Vorgang des Abtastens nennt man »Sampling«. Die Frequenz, mit der die Lautstärke abgetastet wird, heißt dementsprechend »Sampling-Frequenz«. Es läßt sich zeigen, daß es für die einigermaßen naturgetreue Digitalisierung eines Klangs nötig ist, mindestens eine doppelt so hohe Sampling-Frequenz zu verwenden, wie die des Klanges selbst.

Ein solches Sampling ergibt eine Folge von Werten, die dem Schwanken der Amplitude um eine Nullachse entsprechen. Beim Amiga, der diese Werte mit 8-Bit-Genauigkeit verwendet, schwanken die Werte zwischen +127 und -128. Wie bereits oben erwähnt, ist dies eine recht grobe Messung, die aber für viele Zwecke trotzdem völlig ausreicht.

Diese Folge von 8-Bit-Werten (Zahlen zwischen +127 und -128) wird von der Audio-Hardware des Amiga für die Klangausgabe verwendet. Die Zahlen entsprechen dabei der wechselnden Spannung an einem der beiden Lautsprecherausgänge. Diese Spannungsschwankungen setzt der Lautsprecher wieder in Amplitudenschwankungen, und somit in einen Klang um.

Es ist sehr aufwendig, mehrere Minuten Musik in dieser Form zu digitalisieren. Jeder Wert in der Sampling-Folge umfaßt ja 1 Byte (8 Bit). Bei einer Sampling-Frequenz von nur 10 kHz fallen also jede Sekunde 10 KBytes Daten an. In einer Minute also schon 600 KBytes, also fast eine volle Diskette. Man speichert deshalb normalerweise nur eine charakteristische Folge von Werten, die die Kurvenform eines Klangs beschreibt und die sich dann wiederholt. Mit der Beschreibung einer charakteristischen Kurvenform kann man den Klang mit verschiedenen Frequenzen (Tonhöhen) einigermaßen naturgetreu wiedergeben, ohne ein Sampling des Klanges bei exakt dieser Frequenz zu haben. Die Kurvenform wird einfach gestreckt und gestaucht.

Eine einzelne Note hat bei jedem Instrument einen typischen Lautstärkenverlauf (diesmal ist die »echte« Lautstärke gemeint). Zu Beginn eskaliert er schnell zu seiner Maximal-Lautstärke, bleibt dort einen Moment auf einem Plateau stehen und fällt dann langsam wieder ab. Diesen Verlauf nennt man »Hüllkurve«. Die Länge und Steilheit der verschiedenen Plateaus, Steigungen und Gefällen in der Hüllkurve sind von Instrument zu Instrument sehr verschieden und verursachen den typischen Klang.

Die Länge der Plateaus und die Steigungen der Lautstärken-Änderungen müssen bei der Wiedergabe eines Tones angegeben werden. Die Sound-, Hard- und Software ändert dann von sich aus in den wenigen Sekunden, die so ein Ton andauert, die Lautstärke entsprechend.

Hello, I'm the Amiga

Eine Sound-Fähigkeit des Amiga darf man auf keinen Fall vergessen: Die Sprachausgabe. Der Amiga dürfte so ziemlich der erste Computer sein, in dessen standardmäßiger Systemsoftware Sprachsynthese integriert ist. Sie kann in jedem Programm eingesetzt werden und bereitet keinerlei Schwierigkeiten in der Anwendung. In der simpelsten Form übergibt man einfach einen ganz normalen Text an eine bestimmte Routine, die ihn daraufhin in akustischer Form ausgibt.

Die Aussprache variiert frei nach Lust und Laune des Programmierers von tiefem Baß bis zur hellen Frauenstimme. Die Geschwindigkeit der Sprache kann innerhalb weiter Grenzen geändert werden (in weiteren Grenzen, als sinnvoll ist). Und schließlich kann sich die Stimme entweder sehr ausdrucksvoll, mit betonten Worten oder monoton, computerhaft artikulieren.

So steht es zumindest in der Dokumentation. Daß die ganze Angelegenheit zumindest für uns Deutsche nicht ganz so einfach ist, stellt man sehr schnell fest. Die Umsetzung von geschriebenem Text in Sprache orientiert sich an der englischen Sprache.

Ein deutscher Text ergibt manchmal ein völlig unverständliches Kau derwelsch.

Glücklicherweise kann man auch eine Ebene tiefer ansetzen. Die Sprachsynthese-Software des Amiga kann nicht nur Klartext aussprechen, sondern auf Wunsch auch »Phoneme«. Phoneme kennt jeder, der schon einmal in ein Wörterbuch geschaut hat. Es sind die »Urbausteine« der Sprache, elementare Laute, aus denen jedes Wort jeder Sprache zusammensetzbar ist. Macht man sich dann die Mühe, deutsche Wörter gemäß ihren Phonemen einzugeben, ist der Amiga ohne weiteres dazu zu überreden, deutsche Sätze auszusprechen. Einen unverkennbaren amerikanischen Akzent gewöhnt er sich allerdings nicht ab.

Noch ein Wort zur Qualität der Sprache. Mit wenig Mühe kann man die aus dem Amiga kommenden Worte zwar verstehen, die Betonung liegt aber auf »Mühe«. Es hat kaum Sinn, mit Sprache von dieser Qualität Fehlermeldungen auszugeben oder dem Benutzer andere wichtige Informationen mitzuteilen. Mißverständnisse häufen sich dann zu sehr. So werden die Sprach-Fähigkeiten ohne wesentliche Qualitätsverbesserungen wahrscheinlich vorerst auf Spielereien beschränkt bleiben.

Qualität und Geschwindigkeit

Nachdem Sie nun einen ersten Eindruck von Grafik und Sound des Amiga bekommen haben, wollen wir auch noch kurz darauf eingehen, wie sie zustandekommen – vor allem wieso die Amiga-Grafik so schnell ist. Eine allzugenaue Kenntnis der Hardware-Zusammenhänge ist nicht (nicht einmal für die Programmierung) nötig, da einem die Amiga-Benutzerschnittstelle Intuition das meiste der Bit-Fummelei abnimmt. Deswegen nur ein Überblick. Falls Sie allerdings ein Videospiel schreiben wollen, wie es die Welt noch nicht gesehen hat, kommen Sie kaum um eine direkte Manipulation der Grafik-Chips herum. Dann sind allerdings wirklich die Einfälle nötig, die »die Welt noch nicht gesehen hat«.

Qualität und Geschwindigkeit der Amiga-Grafik haben viele Gründe. Zum einen verfügt jeder der drei Zusatzchips im Amiga ungefähr über die Komplexität der CPU M68000, des Hauptprozessors im Amiga. Der Computer erbringt somit, jedenfalls für bestimmte Aufgaben, ungefähr die drei- bis vierfache Rechenleistung, wie ein Computer, der »nur« einen 68000 enthält. Die drei Chips tragen sinnigerweise die Spitznamen »Agnus«, »Denise« und »Paula« (obwohl gelegentlich auch andere Namen zu lesen sind) und bilden logisch einen großen Zusatzchip. Beim heutigen Stand der Technik mußten sie aber auf drei einzelne Chips aufgeteilt werden. Es sind im wesentlichen diese Chips, die die relativ guten Grafik-Fähigkeiten der alten 8-Bit-Ataris ausmachen. Die Philosophien hinter den Atari-Chips und denen des Amiga ähneln einander sehr, nur sind die Amiga-Chips ungleich leistungsfähiger. Der Amiga kann deshalb mit Fug und Recht als ein Nachfolger der alten Ataris bezeichnet werden – eher jedenfalls als die neue Atari ST-Serie.

Arbeitsteilung ist alles

Die drei Chips enthalten logische Untereinheiten, die sich um bestimmte Teilaufgaben kümmern. Diese Untereinheiten werden vom Programm wie ein Coprozessor angesprochen. Das ist ein Prozessor, der parallel zur Arbeit der CPU, Aufgaben erledigen kann. Auf anderen Computern sind zum Beispiel die Chips, die Fließpunkt-Arithmetik erledigen, oft selbständige Coprozessoren. Ein Coprozessor wird von einem Programm aus angesprochen, indem bestimmte Register in den Chips (bestimmte Speicherstellen) mit Werten geladen werden, die ihm sagen, was er tun soll. Genauso wird die Video-Logik des Amiga manipuliert. Indem bestimmte Werte in ihre Register geschrieben werden, die zum Beispiel bestimmen, welche Grafikauflösung der Bildschirm hat, welche Farben in der Palette sind, und so weiter.

Zwei der erwähnten Coprozessoren sind der »Blitter« und der »Copper«. Beide tragen ihren Teil zur schnellen Amiga-Grafik bei. Der Blitter kann vor allen Dingen rechteckige Teile von Bitmaps hin- und herschieben und dabei komplexe Operationen damit ausführen. Zusätzlich zieht er gerade Linien, die entweder »geschlossen« sind oder aus einer beliebigen Folge von Strichen und Punkten bestehen. Schließlich kann er noch ganze Flächen mit einer Farbe füllen – und zwar sehr schnell. Blitter ist ein Kunstwort, das aus »Bit-Block-Transfer« gebildet wurde.

Um eine dieser Operationen auszuführen, müssen die entsprechenden Register des Chips mit Werten versorgt werden. Sie sagen ihm, auf welchen Bit-Maps, Rechtecken und Flächen er arbeiten soll. Nun aktiviert man den Blitter und kann sich bereits anderen Aufgaben zuwenden. Während das eigentliche Programm diese ausführt, arbeitet der Blitter eigenständig weiter. Nach Beendigung meldet er dies dem Programm – auf Wunsch durch einen Interrupt – und wartet auf neue Instruktionen. Auch das ist Multitasking.

Der Copper – der Name steht einfach für Coprozessor – ist ein sehr simpler Mikroprozessor. Er besitzt nur einen sehr eingeschränkten Befehlssatz, der es nicht einmal mit dem alten 6502 aufnehmen kann. Trotzdem ist auch er für einige Aspekte der fixen Amiga-Grafik verantwortlich, besonders für die Mühelosigkeit, mit der sie programmiert werden kann.

Obwohl der Copper ein sehr simpler Mikroprozessor ist, hat er doch die gesamte Amiga-Grafik »in der Hand«. Innerhalb der Zeit, in der der Elektronenstrahl am Bildschirm zirka zwei Zeilen abtastet, kann der Copper fast alle Register der Video-Chips mit neuen Werten versehen. Kleinere Änderungen beanspruchen dadurch die wenige Zeit, die der Strahl braucht, um von einem Bildschirmpunkt zum nächsten zu springen.

Jetzt ist auch verständlich, wie die virtuellen Bildschirme von Intuition zustande kommen. Wenn zwei Screens am Bildschirm zu sehen sind, dann gibt es eine Folge von Copper-Instruktionen, die dafür sorgen, daß in der kurzen Zeit, die vom sichtbaren Ende des einen bis zum Beginn des folgenden, alle Video-Register entsprechend ändern. Auf keinem anderen Computer ist so etwas (virtuelle Bildschirme mit total verschiedenen Grafikeigenschaften) möglich, ohne daß ein Programm nicht schon allein mit diesem Trick vollständig ausgelastet ist. Auf dem Amiga geschieht es völlig mühelos – quasi im Hintergrund und ohne daß ein Programm eingreifen muß.

Die Sound-Hardware des Amiga ist – zumindest aus Programmierer-Sicht gesehen – bei weitem nicht so komplex, wie die Grafik-Hardware. Sie kann im wesentlichen in zwei verschiedenen Modi arbeiten. Der erste Modus benötigt kaum einen Eingriff seitens des eigentlichen Programms, der andere verlangt eine ständige Aufsicht des Programms über die Hardware und hält den Amiga dementsprechend von seinen anderen Aufgaben ab.

Im ersten Modus legt das Programm an einer bestimmten Stelle im Speicher die digitalisierte Wellenform, die Hüllkurve, des zu erzeugenden Tons fest. Anschließend erzeugt das Programm den Ton. Erst danach meldet der Sound-Chip dies dem Programm und verlangt nach neuen Daten.

Im zweiten Modus macht die Sound-Hardware fast gar nichts mehr alleine. Das Programm muß hier die Spannung, die am Lautsprecherausgang anliegt und die Dauer, während der sie konstant gehalten werden soll, direkt in ein Register schreiben. Ist die angegebene

Zeitdauer vorbei, meldet der Sound-Chip dies dem Programm und wartet auf einen neuen Wert. Das eigentliche Programm unterbricht so immer wieder und in sehr kurzen Abständen, da es die Sound-Hardware mit neuen Daten zu versorgen hat.

Gerade ein neuer Computer wie der Amiga lebt und stirbt mit der Software und zusätzlichen Hardware-Produkten. Damit ist nicht so sehr die System-Software gemeint, sondern die kommerziellen Produkte, wie Textverarbeitung, Tabellenkalkulation, Datenbanken. Denn was nützt Ihnen der schönste Computer mit phantastischen Grafik- und Sound-Fähigkeiten, wenn Sie die Textverarbeitung oder Buchhaltung, die Sie eigentlich mit Ihrem neuen Computer erleichtern wollten, nicht bekommen. Deshalb nun ein Überblick über die Software und Hardware, die mit Sicherheit oder sehr hoher Wahrscheinlichkeit schon auf dem Markt, wenn Sie diese Sonderausgabe in Händen halten – zunächst aber wohl fast immer nur in englischen Versionen.

Die Entwicklung neuer Hard- und Software für den Amiga geht – gerade verglichen mit der Markteinführung anderer Rechner – überraschend schnell. Viele der sogenannten »Third Party«-Hersteller sehen offenbar erstaunliche Marktchancen in diesem neuen Gerät und beeilen sich, Produkte auf den Markt zu bringen, um von Anfang an dabei zu sein.

Drum herum

Auf der Hardwareseite gibt es vor allem drei interessante Entwicklungen: die erste Festplatte, das GenLock-Interface und den FrameGrabber. Die englische Firma »Tecmar« stellte bereits eine Erweiterungsbox vor, die in den Expansionport an der Seite des Amiga gesteckt wird, unter anderem eine Speichererweiterung um bis zu 1MByte enthält und den Anschluß einer Festplatte ermöglicht. Eine dazu passende Festplatte wurde auch bereits vorgestellt. Beide Produkte waren schon Anfang Dezember 1985 auf der Amiga-Developer-Konferenz in Eastbourne zu sehen. Ein genaues Lieferdatum wurde nicht angegeben. Die Platte erweckte aber schon nicht mehr den Eindruck eines Prototypen, wenn auch die zugehörige Software noch ein paar kleine Macken hatte und nicht allzu schnell war. Zur deutschen Markteinführung auf der CeBit ist der Amiga aber zweifellos mit Festplatte zu haben. Da die Tecmar-Erweiterungsbox einen Standard-Festplattenanschluß (SASI) enthält, werden sicherlich auch Platten anderer Hersteller bereitstehen. Das GenLock-Interface stammt von Amiga selbst und erlaubt ein Hinterlegen von computergenerierter Grafik mit Bildern aus einer Videoquelle; von einer Kamera, einem Videorecorder oder einer Bildplatte beispielsweise. Das Videobild ersetzt dabei die Hintergrundfarbe (Farbregister 0). Das in Eastbourne gezeigte Interface war ein Prototyp, wird nach Aussagen der Amiga-Entwickler jedoch in Kürze in den USA auf den Markt kommen und soll dort ungefähr 300 Dollar kosten.

Während das GenLock-Interface im wesentlichen nur ein Mischen von extern und im Computer erzeugten Bildern erlaubt, ist mit dem FrameGrabber (zu deutsch ungefähr »Bildfänger«) eine echte Bildverarbeitung möglich. Der FrameGrabber nimmt ein von außen kommendes Videosignal und legt die darin enthaltenen Informationen im Speicher des Computers ab. Er »digitalisiert« es. Dort können sie von einem Programm bearbeitet und dann wieder auf dem Bildschirm dargestellt werden. Mit dem Amiga läuft dieser Vorgang immerhin so fix ab, daß Schwarz-Bilder ebenso schnell wieder auf den Bildschirm gezaubert werden, wie sie von der Videoquelle geliefert werden, und das sind 25 bis 30 Bilder pro Sekunde.

Das waren die wichtigsten Vorstellungen auf der Hardwareseite. Aber auch die Software soll nicht zu kurz kommen. So existieren bereits jetzt zwei Malprogramme für den Amiga auf dem Markt: »GrafiCraft« und »Deluxe Paint«. »GrafiCraft« kommt von Commodore/Amiga selbst und ist als Malprogramm für den Einsteiger einzuschätzen. Es bietet alle wesentlichen Grundfunktionen, die man von einem Malprogramm erwarten kann. Nachteilig wirkt sich natürlich aus, daß lediglich eine Bildschirmseite zur Bearbeitung freisteht und die »Malutensilien« eher dürftig zu nennen sind.

Dafür kann aber das Programm mit 16 oder 32 brillianten Farben aufwarten und verfügt über eine spezielle Funktion, auch einfache Animation auszuprobieren.

»Deluxe Paint« von Electronic Arts, kurz DPaint genannt, ist hingegen mit ziemlicher Sicherheit das leistungsfähigste Malprogramm, das für Hardware, die weniger als 50000 Mark kostet, überhaupt zu haben ist. Es erfüllt alle Wünsche, die ein Zeichenprogramm nur erwecken kann. So kann man »gleichzeitig« zwei getrennte bildschirmgroße Zeichnungen bearbeiten, zwischen diesen rasch hin- und herwechseln und Ausschnitte von einer zur anderen bewegen. Eine Vergrößerung erleichtert Detailarbeiten der beiden bearbeiteten Bilder in mehreren Stufen. »Color register animation«, wie bei GrafiCraft, ist selbstverständlich auch eingebaut. Neben maximal 32 Farben stehen zum Zeichnen natürlich auch eine Vielzahl verschiedener Pinselformen und die allseits beliebte Spraydose zur Verfügung. Jeder beliebige Teil einer Zeichnung kann aber auch als Pinsel verwendet werden, womit die erstaunlichsten Effekte erzielbar sind. Wer das erste Mal sieht, wie ein vielfarbiger Pinsel, der ein Viertel der Bildschirmfläche umfaßt, über die Zeichnung bewegt wird, der wird nie wieder daran zweifeln, daß der Amiga einer der schnellsten Grafikcomputer überhaupt ist.

Weitere Programme, die in Eastbourne gezeigt wurden, waren das Kompositions- und Synthesizer-Programm »MusicCraft«, die Textverarbeitung »TextCraft« und diverse Spiele, die teilweise in Prerelease-Versionen vorlagen. »MusicCraft« zeigte eine erstaunliche Klangqualität, wenn man die Programmausgabe über eine Stereoanlage schickte.

Die Software macht's

Für den professionellen Einsatz muß man wohl auf »Deluxe Music Construction Set« von Electronic Arts warten, das in einer Macintosh-Version bereits erhältlich ist und dort (auch für Profis) nur sehr wenige Wünsche offen läßt. Die Klangqualität des Amiga erreicht der Mac allerdings nicht ganz. »Deluxe Music Construction Set« ist etwa ab dem Frühjahr in den USA auch für den Amiga zu haben und steht somit wohlrechtzeitig zur deutschen Markteinführung des Rechners zur Verfügung. Eine ausführliche Beschreibung finden Sie in dieser Ausgabe.

Das Programm »TextCraft«, das von Commodore/Amiga selbst vertrieben wird, erwies sich hingegen als etwas enttäuschend. Es ist – dem ersten Eindruck nach – kaum für anspruchsvollere Aufgaben einsetzbar. Lediglich für das Erstellen kleinere Texte oder von Briefen ist es akzeptabel, ansonsten recht umständlich zu bedienen und nutzt kaum die außerordentlichen Fähigkeiten, die der Amiga anbietet.

Die Stärken des Amiga sind recht deutlich: Seine sehr gute und schnelle Farbgrafik, die Animationsmöglichkeiten und das Multitasking. Gerade für den Programmierer, der sich für diese Gebiete interessiert, ist der Amiga eine sehr reizvolle und im Moment ziemlich konkurrenzlose Maschine.

Auch Anwender, die mit dem Medium Video oder Bildplatte zusammenarbeiten oder Bildverarbeitung betreiben, sind mit dem Amiga bestimmt gut bedient. Dies gilt natürlich auch für Hersteller, die solche Produkte entwickeln wollen! (Markus Breuer/hb)

Der QL wird zum PC

MP steht weder für »Member of Parliament« noch für »Maschinenpistole«, sondern für »Micro Peripherals Ltd«. Diese Firma hat eine Disketten-Station auf den Markt gebracht, die den Großen von Sinclair zum Personal Computer werden läßt.

Der Sinclair-QL bietet den preiswertesten Einstieg in die neue Klasse von Computern, die auf der Basis der Motorola-68000-Familie aufbauen. Zusammen mit der im Preis inbegriffenen professionellen Software der Firma Psion, bietet dieser Computer ein bislang nicht dagewesenes Preis/Leistungsverhältnis.

In der Erwartung, daß sich der QL gut verkaufen wird, haben sich eine ganze Reihe von Soft- und Hardware-Herstellern auf den QL gestürzt. Eines der nun auf dem englischen Markt erschienenen Produkte wollen wir hier genauer unter die Lupe nehmen. Eine geeignete Disketten-Station kann den QL zu einem vollwertigen Profisystem machen, ohne dafür einen hohen Preis zu fordern. Von Sinclairs Seite wurde mittlerweile das Produkt von Micro Peripherals auserkoren, unter dem eigenen Logo verkauft zu werden. Dies ist nicht verwunderlich, ist das Micro Peripherals-Laufwerk doch genau auf die Bedürfnisse des QL zugeschnitten, und bietet dem Benutzer zusätzlich außerordentlich nützliche Software an

Doch zunächst zum eigentlichen Gerät. Es ist ein sehr kleines, dafür aber nicht zu leichtes Kästchen, das in dezentem Schwarz gut zum Äußeren des QL paßt. Dazu gehört natürlich ein Controller, der in den seitlichen Expansion-Port des QL eingeschoben wird. Controller und Laufwerk sind durch ein etwa ein Meter langes Flachbandkabel miteinander verbunden. Das schafft genug »Bewegungsfreiheit« auf dem Schreibtisch.

Das Laufwerk erhält seinen Strom durch ein eigenes Netzteil; auch hier ist ein langes Verbindungskabel vorhanden. Um den Betrieb aufzunehmen, wird zunächst der Controller in den seitlichen Port geschoben. Anschließend kann das 34adrige Kabel mit dem Laufwerk verbunden werden (wobei dies nicht unbedingt das von Micro Peripherals sein muß). Allerdings empfiehlt die Firma aus recht verständlichen Gründen zur Benutzung ihre eigenen Produkte.

Format nach Wahl

Insgesamt können bis zu vier Laufwerke gleichzeitig verwendet werden, wobei die Wahl zwischen 3-, 3½- und 5¼-Zoll-Laufwerken besteht. Das Micro Peripheral-Laufwerk ist ein 3½-Zoll-Gerät und damit mit dem zukünftigen Standard konform.

Formatiert haben die Disketten eine Kapazität von 720 KByte (double sided und double density). In dem Zustand, in dem die Laufwerke geliefert werden, kann man sie mit dem Namen »fdk« per Software ansprechen. Gefällt der Name »fdk« nicht, so kann man dies leicht ändern, indem man den direkten Befehl »VSET name« eingibt. Von nun an hören die Laufwerke auf den selbst definierten Namen.

Ist man der Microdrives überdrüssig, kann man durch den Befehl »MSET« die Disketten-Laufwerke auf »MDV« hören lassen. In diesem Fall sind die Microdrives nicht mehr zu benutzen, bis man den Befehl »FSET« eingibt, der wieder den ursprünglichen Zustand herstellt. Der Vorteil dieser Befehle liegt auf der Hand: Möchte man zum Beispiel die Psion-Software benutzen und seine Programme und Datenfiles auf Disketten sichern, so ist dieses ohne irgendwelche Probleme möglich. Die Laufwerke lassen den QL im Glauben, er habe es mit Microdrives zu tun.

Der QL-Kenner wird nun sagen, das ist ja ganz schön, aber ein automatisches »Booten« eines Programms muß noch immer von Microdrives erfolgen. Selbst daran ist bei den Laufwerken gedacht. Am Controller sind kleine Verbindungsbrücken angebracht, die die eben besprochenen Befehle auch per Hardware durchführen. Somit kann mit der entsprechenden Stellung einer dieser steckbaren Brücken jedes Programm direkt von Diskette gebootet werden.

Das einzige Problem bieten hier, natürlich wie immer, die kopiergeschützten Programme. Sie können zwar leicht auf Diskette transferiert werden, müssen jedoch zum Start immer noch ein Paßwort von dem entsprechenden Cartridge lesen. Dennoch ist auch in diesem Fall eine erhebliche Zeitersparnis vorhanden. Neben den eben besprochenen Befehlen, kann mit den Befehlen »DPUT« und »DGET« ein beliebiger Sektor mit Daten gefüllt, beziehungsweise von diesem Sektor Daten gelesen werden.

Nach mehrwöchigem Betrieb ist eindeutig festzustellen, daß die Laufwerke sehr zuverlässig und schnell arbeiten. Die Zugriffszeiten sind sehr kurz und als Besitzer der Laufwerke ändert man für längere Programme durchaus seinen Programmierstil. Programme werden nun nach Belieben in kleinere Teilprogramme unterteilt und bei Bedarf aus einem jeweiligen Menü abgerufen. Durch das Multitasking des QL fallen die Wartezeiten beim Übergang von einem Programm zum anderen, selbst bei zwischenzeitlichem Sichern von Daten, nicht mehr ins Gewicht.

Das naheliegendste Programm ist natürlich ein Steuerprogramm für die Psion-Software, aus welchem die Programme Archive, Abacus, Easel und Quill aufgerufen werden können. Dabei ist es leicht so einzurichten, daß nach der Arbeit mit einem Programm wieder in das Hauptmenü zurückzukehren und in einem anderen Programm, zum Beispiel nach dem »Export« von Daten, weiterzuarbeiten ist.

Utilities gratis

Mit dem Laufwerk wird eine Diskette voller Utilities geliefert, die sich als sehr nützlich erweisen. Unter den zur Verfügung gestellten Programmen befinden sich zum Beispiel:
- eine Routine, mit der bis zu neun Fenster gleichzeitig erzeugt und jederzeit verändert werden können
- eine »Back-Up« Routine, mit der beliebig zwischen Microdrives und Laufwerken hin und her kopiert werden kann
- ein Programm, mit dem in jedem File ein beliebiger String gegen einen anderen ausgetauscht werden kann. So ist es möglich, zum Beispiel in den Psion-Programmen »MDV« durch »FDK« oder in den Anleitungen das Wort »Microdrive« durch »Floppydisc« zu ersetzen.
- eine Routine, die jedes Disketten- oder Microdrivefile inklusive Header auf den Bildschirm bringt.
- eine »Screen-Dump« Routine.

Das Laufwerk mit Controller kostet knapp unter 1000 Mark, ein zweites Laufwerk knapp 600 Mark. Der englische QL wird hierzulande mittlerweile mit einem Laufwerk und Farbmonitor für 2300 Mark verkauft. Bedenkt man, daß darin ein Paket mit hervorragender Software enthalten ist, so liegt in dieser Kombination ein Preis/Leistungsverhältnis vor, das die Konkurrenz bislang noch nicht unterbieten konnte.

(P. C. Bosetti/Manfred Kotting/hb)

Die richtige Wahl

Wir haben für Sie die für eine Kaufentscheidung wesentlichen Kriterien der 68000-Computer gegenübergestellt.

Vielleicht haben auch Sie sich schon mit dem Gedanken auseinandergesetzt, Ihren Arbeitsplatz um einen der 16-Bit-Super-Computer zu bereichern? Diese Maschinen überragen in ihren Leistungen die Heim- und die meisten Personal Computer weit. Ihrem Einsatz steht, dank fortschreitender Entwicklung von Hard- und Software, nichts im Wege. Doch ohne Überblick fällt die Entscheidung schwer.

Ein Computer dieser Preisklasse sollte optimal auf die Bedürfnisse des Anwenders abgestimmt sein. Denn ob Sie einen Grafikspezialisten oder einen universellen Rechenknecht von der Textverarbeitung bis zur Dateiverwaltung brauchen, ist letztendlich Ihre Entscheidung. Und genau hier soll unsere Übersicht ansetzen: Die Auswahl eines Computers dieser Klasse wird ausschließlich von seinen Leistungsdaten bestimmt, die wir Ihnen in Form einer Tabelle anbieten. (og)

Merkmale	Gepard-PC	Commodore Amiga	Atari 520 ST+	Stride	Apple Macintosh	Sinclair QL
Hauptprozessor	68000	68000	68000	68000	68000	68008
RAM	512 KByte	256 KByte	1MByte	1MByte	128 KByte oder 512 KByte	128 KByte 640 KByte
Erweiterbar	1MByte	512KByte	-	12 MByte	1 2 4 MByte	-
Extern erweiterbar	-	8 5 MByte	4 MByte	-	-	-
ROM	16 KByte	192 KByte	192 KByte	16 KByte	64 KByte	48 KByte
Taktfrequenz	10 MHz	7 159 MHz	8 MHz	12 MHz	7,8336 MHz	8 MHz
Bildschirmauflösung monochrom	1000x700	640x400	640x400	externe	512x342	
Punkte Farben	280x210/256, erweiterbar	320x400 32	320x200/16	Terminals oder 784x325	-	256x256 8 512x256 4
Sprites	keine	8	keine	keine	keine	keine
Farben maximal	256	4096	512	-	schwarz/weiß	8
Sound	2 Kanäle	4 Kanäle	3 Kanäle 1 Rauschgenerator	möglich	4 Kanäle	Beeper
Schnittstellen	2xCentronics RS232 Diskettenlaufwerk Festplatte 16 Slots Maus/Joystick Stereosound	Centronics RS232 Diskettenlaufwerk Festplatte Maus Joystick Systembus Stereosound	Centronics RS232 Diskettenlaufwerk Festplatte Midi ROM-Cartridge Maus/Joystick	Centronics RS232 Diskettenlaufwerk Festplatte	RS232 RS442A Maus	2xRS232 Network-Anschluß Joystick Systembus
Monitorsignale	RGB analog RGB digital PAL-Composite	RGB analog RGB digital NTSC Composite TV Anschluß	RGB analog	-	-	RGB
Diskettengröße	3½ Zoll 5¼ Zoll	3½ Zoll 5¼ Zoll	3½ Zoll	5¼ Zoll	3½ Zoll	2 x Microdrives
Diskettenkapazität	800 KByte	880 KByte	360 720 KByte	640 KByte	400 KByte	100 KByte
Festplattenkapazität				10 448 MByte		
Netzwerkfähig				max 64 Computer		max 64 Computer
Betriebssystem	GDOS, CP/M-68K	AmigaDOS	TOS	UCSD-Pascal RM-Cos. CP/M 68K und weitere		QDOS
Multitaskingfähig	nein	ja	nein	ja	-	ja
Benutzeroberfläche	Betriebssystem	Intuition	GEM	Betriebssystem	Finder	-
Tastatur	separat	separat	integriert	separat	separat	integriert
Tastenzahl	95	89	94	vom Terminal abhängig	59	65
Maße (BxHxT) in cm	36x40x16	45x11x34	47x6x24	Diverse	25x34x28	47x4 5x13 5
Lieferumfang: Hardware	integrierte Disk	12 Zoll Farbmonitor integrierte Disk Maus	12 Zoll Monitor externe Disk Maus		9 Zoll Monitor integrierte Disk Maus	QDOS Super Basic
Software	GDOS Paradise Modula 2 Systemeditor Debugging-Tool Gepcalc Gepstar	AmigaDOS Basic Tutorial Kaleidoscope Voice Synthesis	TOS Basic Logo GEM-Write GEM-Paint	Nach Wahl	Macwrite Macpaint	Abacus Archive Easel Quill
Preis	Textpaket: 5798 Mark Grafikpaket: 6798 Mark	etwa 5900 Mark + MwSt	2998 Mark	12 000 Mark bis 200 000 Mark	8265 Mark	998 Mark

Vergleichstabelle der Leistungskriterien von 16-Bit-Computern mit 68000-CPU

Ein persönlicher Computer – Atari ST

Der Atari ST vereinigt die besten Eigenschaften von Gebrauchs-, Heim-, Büro-, Spiel- und Wissenschafts-Computern als idealer Konsumartikel.

Personal Computer heißt nicht »Computer für das Personal«, sondern eigentlich »persönlicher Computer«. Das heißt, ein Computer, der von einer einzelnen Person benutzt wird, oder besser gesagt, der einer Person mit seiner ganzen Leistung voll zur Verfügung steht, der auch physisch am Arbeitsplatz installiert ist. Daher auch der Begriff »Arbeitsplatzrechner« – aber das klingt so deutsch-moralisch, fast wie »Bürocomputer«.

Als ich Anfang '83 die Lisa von Apple kennenlernte (sie schrieb sich ja damals auch in den Anzeigen »von«), war ich so überwältigt, daß ich meine Eindrücke nur in ein nun schon historisches Lisa-Poem kleiden konnte. Ich hasse Betriebssysteme wie CP/M oder MS-DOS oder – gar – Unix. Trotzdem haben mich Apples Computer seither nicht mehr besonders interessiert. Ich habe daran viel über Benutzeroberflächen gelernt, habe im Laufe von 1½ Jahren in meiner Freizeit ein Programm entwickelt, das manche dieser Techniken verwendet – weil es mich interessierte, was dahintersteckt, und weil ich herausfinden wollte, ob ich es schaffe, so etwas zu machen. Gerade dazu war aber eine Lisa oder ein Macintosh kaum zu gebrauchen. Für mich muß ein Computer ein universales Betriebssystem haben, sich programmieren lassen. Ich bin, obwohl ich oft betone, ein Anwender zu sein, in meiner Seele ein Programmierer, weil ich ein schöpferischer Mensch bin, der Dinge in die Welt setzen will, die vorher noch nicht da waren. Zufällig geschah es, daß ich Computer als Ausdrucksmedium dafür entdeckte. Indem man sie programmiert, schafft man etwas, das real existiert (das Programm). Zugleich sind Programme etwas, das selbst aktiv ist – sie sind wie eigenständig handelnd gewordene Gedanken des Programmierers, des Menschen.

Apple hat die Lisa und den Mac so gestaltet, daß man diese Computer nur »gebrauchen« sollte – etwa wie ein Auto –, um im Büro Textverarbeitung, Tabellenkalkulation und dergleichen zu betreiben – Dinge, die mich herzlich wenig interessieren. Dazu kam, daß diese Computer durch nicht-standardmäßige Schnittstellen und Softwarekonzepte vom Rest der Welt hermetisch abgeriegelt waren. Außerdem waren sie ziemlich teuer, gemessen an Computern, die man programmieren kann. Naturlich kann man den Mac auch programmieren, aber es ist vergleichsweise sehr kompliziert – in merkwürdigem Gegensatz zum einfachen Gebrauch. Heute gibt es mehr Möglichkeiten zur Programmierung eines Mac als damals bei der Lisa (insofern ist die Situation des Atari radikal anders, denn der kam mit Zugangsmöglichkeiten für den Programmierer auf die Welt). Aber als ich mit einem mir gut bekannten Hacker, der seit längerem einen Mac besitzt, darüber am Telefon sprach, ob er wohl ein Programm mit serieller Ein-/Ausgabe an den Mac anpassen könnte und von ihm nur ein resigniertes »Das ist sehr schwierig, da wüßte ich erst mal gar nicht, wie ich das machen soll« hörte, war mir klar: Der Mac ist kein Programmierer-Computer. Auf dem Atari ST gelang es mir (obwohl ich partout kein Systemprogrammierer bin) innerhalb von 14 Tagen, drei Versionen meines Bagger-Programms zum Funktionieren zu bringen. Dabei hatte ich lediglich mit einem unausgereiften Editor zu kämpfen. (Das Bagger-Programm dient der Übertragung von Dateien aller Art zwischen verschiedenen Betriebssystemen und Computern über die serielle Schnittstelle. Sie können es für den Atari und IBM-PC und eventuell einige andere auf Diskette bekommen.)

Die Konkurrenz vollzog Apples Revolution

So blieb die prophezeite Revolution durch Apple zunächst aus (daß alles seine Reifezeit haben mußte, war uns natürlich damals auch klar). Mir scheint, daß heute die Konkurrenten, wenn auch mit dreijähriger Verspätung, die zwei Hauptfehler von Apple vermeiden: Ihre Computer sind billig statt teuer, und sie beruhen auf Standardkonzepten statt Insellösungen. Möchte man von einem Atari einen grafischen Ausdruck vom Bildschirm oder einer grafischen Datei haben, nimmt man einfach den am IBM-PC angeschlossenen Matrixdrucker (man verwendet sogar das gleiche Kabel), schließt es an den Atari an und – es geht. Man braucht keinen Spezialdrucker, kein exklusives Kabel.

Es ist auch versucht worden, Benutzeroberflächen wie die des Mac auf einen IBM-PC zu übertragen – aber das scheiterte immer daran, daß diese Computer zu wenig leistungsfähig sind.

IBM hatte einige Jahre die Welt in seinem eisig blauen Griff und zwang uns eine Technologie auf, die durch IBM zwar »Industriestandard« wurde, die aber nicht Stand(ard) der Technik, geschweige denn High Tech (Hochleistungs-Technik) war. Trotzdem befinden sich nun Computer auf dem Vormarsch, deren Konzepte dem Können und Wissen und den Ideen unserer Zeit entsprechen.

Der Atari besitzt ein in seiner Struktur konventionelles Betriebssystem (TOS, das an der Oberfläche MS-DOS sehr ähnelt), verwendet Standardschnittstellen (RS232C, Centronics, MIDI-Interface), bietet zum Drittel des (einstigen) Preises eines IBM-PC eine höhere Leistung als ein AT. Der Atari arbeitet im Vergleich zum IBM-PC drei bis sechsmal schneller, er kann (vom Prozessor her) 16mal soviel Speicher adressieren, und er besitzt mit GEM alle Möglichkeiten, um eine symbolische Benutzeroberfläche für »Gebrauchsprogramme« herzustellen (Graphic Environment Manager). GEM setzt auf TOS auf, GEM ist das Mittel, um die grafisch-symbolische Benutzeroberfläche herzustellen. Damit vereint der Atari das Beste aus beiden Welten: Die standardisierten Programmiermöglichkeiten für den Programmierer/Entwickler und die Anwendungshilfen für den reinen Benutzer, der nichts mit dem Computer als solchem im Sinn hat, sondern der nur seine Sache getan haben will.

Die Gemeinheit ist, daß wir – vielleicht abgesehen vom Preis – das alles im IBM-PC auch schon hätten haben können, denn der Motorola 68000 war (Lisa, Sage/Stride) damals durchaus schon bekannt. Der Industriestandard könnte heute also zehnmal so leistungsfähig sein, als er dank IBM ist. Aber ohne IBM hätten wir keinen Standard. Erinnern wir uns doch einmal an

das absolute Durcheinander vor der Zeitenwende – Commodore, Apple, Tandy – keiner dieser Computer konnte mit einem der anderen, meistens nicht einmal, wenn er vom selben Hersteller war. Und der Atari zehrt ebenfalls von IBM – er verwendet nicht nur dieselben Stecker, sondern auch (fast) denselben Zeichensatz sowie ein Betriebssystem mit sehr ähnlicher Oberfläche wie der PC. Er ist eben nur moderner.

Ein befreundeter Sprachwissenschaftler, der selbst für seine Hochschultätigkeit Software aus dem Bereich KI (künstliche Intelligenz) und Sprachanalyse entwickelt, arbeitete bisher mit demselben Z80 wie ich (einem angejahrten Bürocomputer). An der Universität wurden Olivetti M 24 angeschafft (wegen IBM), er selbst aber kaufte sich jüngst zu meiner Überraschung, aber auch Freude, einen Atari ST. Wir paßten das UCSD-System an, mit dem sich der IBM-PC wacker abrackert – leider läuft mein auf dem Sage (Vorläufer von Stride) entwickeltes Grafikprogramm auf dem PC so langsam, daß es überhaupt nicht mehr zu gebrauchen ist. Auf dem Atari 520 ST mit RAM-Disk läuft UCSD so, daß man nur noch an Turbo-Pascal denken kann. Und Turbo-Pascal – müßte demnächst praktisch mit Lichtgeschwindigkeit arbeiten, wenn es eine 68000er-Version gibt.

Gewiß hätte man Anfang 1983 von Apple nicht verlangen können, die Lisa um 3000 Mark zu verkaufen – es lag auch, aus unternehmerischer Sicht, dafür kein Grund vor, denn es gab keine Konkurrenz für einen solchen Benutzercomputer. MS-DOS war und ist allenfalls ein »verschlimmbessertes« CP/M – es kann zwar technisch weit mehr, aber dem Benutzer gegenüber bietet es allenfalls kosmetische Veränderungen im Vergleich zu CP/M. Andererseits ist die GEM-Software noch nicht so ausgereift wie die des Mac, doch ist Software ja etwas Lebendiges, das sich weiterentwickeln kann. Eigentlich ist es auch ein Vorzug, daß GEM vergleichsweise einfacher ist als das Lisa-System. Vergleicht man GEM mit der Mac-Umgebung, so hapert es häufig nur am richtigen Einsatz der Fähigkeiten des Atari. Manche Programme der Gründerzeit sind einfach in der Benutzerführung schlecht gestaltet. In einigen Punkten erkennt man auch Verbesserungen (siehe Vergleich der Benutzeroberflächen).

Der Atari bietet keineswegs innovative Technik (außer vielleicht in produktionstechnischer Hinsicht), denn zum Zeitpunkt seines Erscheinens ist ein Computer mit einer bei 8 MHz betriebenen 68000-CPU eher konventionell (wie mein Sage, der zwei Computerjahre auf dem Buckel hat). Aber es ist die erprobte moderne Technik, und kein Anachronismus wie der 8088 des IBM-PC, von dem heute vergleichsweise unbedeutenden Chip-Schöpfer Intel, der ohne IBM wohl nur noch im industriellen Steuerungsbereich von Bedeutung wäre.

Wie Hunde haben Computer ein viel, viel kürzeres Leben als Menschen, mit zwei Jahren sind sie im »besten Alter«, wie man das beschönigend nennt, mit vier Jahren sind sie schon ältere Herren respektive Damen, und mit rund sechs Jahren Greise, die bei besonders computerfreundlichen Herrschaften noch ein Gnadenbrot genießen, weil sie so lange treue Dienste geleistet haben – und weil es immer noch manches gibt, in dem die Grünschnäbel ihnen nichts vormachen können.

Spitzentechnologie bietet etwa der Stride mit 12 MHz, 12 Megabyte Speicherausbau, Disk-Cache, Multiuser-Betriebssystem und Netzwerk; innovativ wäre ein Computer mit dem National 32000; aber Spitzentechnologie ist teuer – zu teuer für ein Massenprodukt.

Sicherlich ist der Amiga in vieler Hinsicht mehr High Tech als der Atari, fragt sich nur auf lange Sicht, ob sich hier Wollen und Können genügend ideal vereinen. Der Atari hat, was er braucht, und basta. Er ist, ob man das Wort schön findet oder nicht, in ähnlicher Weise ein Volkscomputer wie Turbo-Pascal ein Volkscompiler war und ist.

Traumcomputer mit Standard-Schnittstellen

Früher träumte ich immer von einem erschwinglichen Computer, der – ohne die bei Spiel- und Heimcomputern üblichen Abstriche – neben standardmäßigen und professionellen Konzepten auch eine Hochleistungs-Grafik enthalten sollte. Apple baute diesen Computer nicht, aber Atari tat es. Wie der Amiga hier einzuordnen ist, bleibt abzuwarten. Ich hatte zum Zeitpunkt, als diese Ausgabe entstand, keine Gelegenheit, ernsthaft mit ihm zu arbeiten.

Ich finde, Computer sollten nicht wie Kühlschränke sein, bei denen der Einsatz des einen Modells nicht in ein anderes hineinpaßt – das widerspricht völlig dem Konzept des Computers als universaler Maschine, die alle anderen Maschinen imitieren kann. Apple entspricht diesem Wunschbild ganz und gar nicht, der Atari schon viel mehr.

Als ich auf der Hannover-Messe 1985 vom Geschäftsführer eines Verlages gefragt wurde, ob ich mir einen Atari kaufen würde, fiel meine Antwort sehr zurückhaltend aus. Ich wußte zu diesem Zeitpunkt so gut wie nichts über TOS, es gab praktisch keine Software und keine Programmiersprachen – was hätte ich mit einem Atari tun sollen, den ich nur vom Hörensagen und von Messeständen kannte? Ein wenig steckten mir noch die eigentlich frustrierenden Langzeiterfahrungen mit Apple in den Knochen. Heute fiele die Antwort über den Atari ganz anders aus, und ich würde sagen: »Also, gegen meinen Sage tausche ich ihn nicht ein. Aber ich könnte ihn mir dazukaufen. Zum Beispiel ist mir für das tägliche Arbeiten die Tastatur zu schlecht – obwohl es eine gut ausgestattete Standardtastatur ist, sehr im Gegensatz zu Apples »Mäuseklavier«. Es zeichnet sich ab, daß der Atari ein Computer ist, für den ich gerne Software und Artikel mache – wenn auch nicht immer auf ihm. Dazu verwende ich teilweise den Sage, teilweise sogar noch meinen alten Z80. Immerhin begann ich ein kleines Projekt der ersten Stunde (die Entwicklung des »Superfilters«, ein Zusatzprogramm für den vorher erwähnten Bagger, das allerlei Textformate in allerlei andere Textformate überträgt) sogar auf dem Z80-Rechner; wechselte dann aber, weil das Programm technisch zu umfangreich wurde, auf den Emulator auf dem Atari (auf dem Emulator hat CP/M mehr Speicherplatz als im wirklichen Leben), um es zuletzt an Modula unter TOS anzupassen (der Editor dieses Modula-Compilers war mir noch zu unausgereift und fehlerträchtig).

Meine zurückhaltende Einstellung war sicherlich auch durch die mir verhaßte heimcomputerhafte Aufmachung mit »zigundzwanzig« Kabeln, Netzteilen und Lose-Rumsteh-Floppys begründet, durch die auf der Messe gezeigten verstörten Farbmonitore ebenso wie Ataris Image als Produzent elektronischen Spiel- und Blendwerkes. Ich irrte – der Atari ST ist ein Low-Cost-Proficomputer. Was man an seiner Tastatur eingespart hat, wird mehr als aufgewogen durch die Qualität des Schwarzweißbildschirmes. Noch besser wäre freilich eine ebensogute Tastatur dazu – aber das ist Mechanik, und Mechanik ist und bleibt teuer.

Der Atari hat einen großen Vorzug, der ihm im Überlebenskampf und dem Ringen um Marktanteile der Hersteller entscheidend helfen dürfte: Er ist, nach dem heutigen Stand der Beobachtungen, ein Computer für Benutzer und Hacker gleichermaßen – und zumindest als Zielmaschine, auch für professionelle Entwickler. Mit einem Atari kann man auch als Schüler Software entwickeln, die moderne Computerpower verlangt.

(le)

Die Computerkultur – Barbarei oder Höherentwicklung?

Ungruen hat seit ein paar Wochen ein neues Spielzeug. Es ist ein ganz besonderes Spielzeug – er führt es stolz nicht nur seinen Freunden (größtenteils Computerfachleuten) vor, sondern mit mindestens ebenso viel Freude den Kindern seiner Besucher. Und es ist ein Spielzeug, das fünfjährige kleine Mädchen (und Jungs) ebenso fasziniert wie Studenten, die gerade einen Compiler geschrieben haben.

Die Szene: Ungruens »Praxis« (so nennen wir sein Labor, weil er darin aus unerfindlichen Gründen immer einen weißen Kittel trägt – als ob er sich am Computer den Bauch schmutzig machen könnte), darin mindestens vier bis fünf Computer, davon drei eingeschaltet, ein Drucker, Plotter und ähnliches, sehr viel herumliegendes Notizpapier, Stapel von Büchern und Zeitschriften – unwegsames Gelände!

Die Personen: Er selbst (U), Achim (mein Freund, A) und ich, Maja (M).

M: Können Sie mir mal in kurzen Worten erklären, was das hier ist ?

U: Nein.

M: – Äh, ich dachte...

U: Schauen Sie, das ist mit so einem Computer wie mit, wie mit... das wollte ich jetzt nicht sagen.

M: Was wollten Sie sagen?

U: Nicht sagen! Wie mit der Liebe – man sollte darüber nicht Bücher lesen oder Filme anschauen, man sollte sie machen.

M: Gut, ich verstehe, das ist etwas, das man erleben sollte...

U: Genau. Schreiben Sie – sagen Sie Ihren Lesern, sie sollen sich einen Nachmittag Zeit nehmen und statt ins Kino zu gehen, verschiedene Geschäfte besuchen, sich dort einen Apple Macintosh, einen Atari ST oder einen Amiga vorführen lassen. Und nehmen Sie Ihre Freundin oder Ihren Freund mit. Und die Kinder, wenn Sie welche haben.

M: Mein Bruder – er ist jetzt 12 und gerade von Basic zu Pascal aufgestiegen, rümpft darüber die Nase. Er meint, das wäre für Kleinkinder und – äh, Leute über 30, die nicht Bescheid wissen.

A: Tatsächlich hört man Hacker und Computerspezialisten oft abfällig über diese Art Benutzeroberfläche reden. Sie glauben, das entmündige den Menschen, degradiere ihn zu einem Analphabeten, der nicht bis drei zählen kann.

U: Man kann diesen Standpunkt durchaus verstehen – aber dies sind nicht die Leute, für die Computer gemacht werden. Computer sind ja nicht dazu da, daß gewisse Menschen ihr Vergnügen darin finden, auf ihnen immer raffiniertere und kompliziertere Programme zu schreiben. Ich muß hinzufügen: Computer sind wohl erfunden und entwickelt worden, weil gewisse Leute daran Vergnügen finden... aber eigentlich sind Computer doch nicht dazu da, daß jemand, der Computer interessant findet, mit ihnen rumspielt. Computer sind, wie Autos, dazu da, etwas zu erledigen, was ein Anliegen von Menschen ist, die nichts mit Computern oder Autobasteln im Sinn haben.

A: Es gibt aber auch Rennfahrer, Autobastler, jugendliche Autofreaks...

U: Natürlich. Und es soll und wird auch weiterhin Computerfreaks geben. Man benutzt aber Autos auch ohne sich dafür zu interessieren, wie ein Viertaktmotor funktioniert – oder können Sie mir das genau erklären?

M: Äh – nein. Aber ich habe eine ganze Menge Fahrstunden gebraucht, bis ich fahren konnte...

U: Sehen Sie – ich habe mal gesagt, alle Programme sind Spiele. Was man bisher erlebt hat, wenn man es mit einem Computer zu tun bekam, waren die aberwitzig konstruierten Spiele der Betriebssystem-Schreiber und Softwaretüftler. Wenn ich von meinem Computer die Adresse von Herrn Maier wissen will, interessiert es mich doch nicht, daß die in einem File steht, und der in einem Subdirectory, und daß ich das erst anwählen muß, oder daß ich eine ganz neu erfundene Spielsprache benutzen muß, um den Computer das zu fragen. Ich will nicht wissen, was ein File ist, wie ein gültiger Suchpfad in einem Directory formuliert wird – ich will nur wissen, wo Herr Maier wohnt. Früher habe ich das im Adreßbuch nachgeschlagen und mußte dazu zwar das Alphabet können (das kann jetzt der Computer), aber ich mußte keinen Zauberspruch aufsagen, damit sich die Seiten öffnen.

A: Ist nicht der Umgang mit der Maus und diesen Fensterchen da und Bildchen, die man verschieben kann, erst recht ein Spiel?

U: Selbstverständlich. Aber ein Spiel, das jeder versteht, weil es von visuellen Begriffen ausgeht, die wir mit alltäglichen Dingen verbinden. Die Gestaltung dieser Spiele (Benutzeroberflächen, technisch gesprochen) hat sehr viel mit Wahrnehmungspsychologie zu tun, mit der Art, wie wir Zusammenhänge und Abläufe, also Gestalten und Kausalität, begreifen und wie wir handeln. Eine gut gestaltete Benutzeroberfläche ist keine Barriere vor dem Gebrauch des Computers, sondern sie hat spielerischen Aufforderungscharakter. Man wird eingeladen, etwas Offensichtliches zu tun.

Es ist sehr eindrucksvoll, ein kleines Kind zu beobachten, das lernt, mit so einem Computer umzugehen. Ich habe immer das Gefühl, daß ich es in seinem Kopf ticken und einschnappen höre – alles, was sich auf dem Bildschirm abspielt, scheint für das Kind offensichtlich und einsichtig zu sein – es entspricht den menschlichen Erwartungen und den Mechanismen, durch die wir die Welt begreifen. Die Leute, die all das erfunden haben – bei Xerox PARC (Palo Alto Research Center) und bei Apple, die haben einfach darüber nachgedacht, wie der Mensch die Welt sieht. Wir zergliedern die Welt in Gegenstände, die wir unterscheiden können, und wenn wir mit einem Gegenstand etwas tun wollen, dann fassen wir ihn an und machen mit ihm etwas. Nun können Computer viel leichter mit Grafik als mit Sprache umgehen. Und sie können mit objektorientierten Konzepten umgehen – das heißt, sie können bestimmte innere Zustände wie Sachen darstellen, die man anfassen kann (mit der Maus).

A: Mir kommt es immer vor, als ob Erwachsene mehr Schwierigkeiten haben...

U: Ja, das hat verschiedene Gründe. Die Erwachsenen sind in ihren Erwartungen nicht mehr so offen, ich würde sagen, ihre Haltung ist nicht mehr abstrakt und naiv genug. Sie glauben nicht, daß man ein Kästchen auf einem Bildschirm anfassen und umherschieben kann, sie glauben nicht mehr an die Magie des Deutens, der symbolischen Handlung. Ein Kind ist sofort bereit, zu glauben, daß etwas verschwindet, das man in den gemalten Papierkorb wirft. Ein Erwachsener sieht zwar den Papierkorb oder die Mülltonne auf dem Bildschirm, aber er ist nicht kreativ genug, ihren Zweck zu begreifen – bis man es ihm erklärt. Der Erwachsene glaubt bis zuletzt, er müsse irgendeinen Geheimcode herausfinden und diesen auf der Tastatur mühsam einbuchstabieren, damit der Computer irgendetwas tut. Und wenn man ihm diesen Geheimcode sagt, hält er dieses Wissen für seine Kompetenz – jedes Kind lacht darüber.

M: Haben dann nicht doch die Kritiker recht, die sagen, diese Bildchen-Sprache sei eines qualifizierten Erwachsenen unwürdig? Und wäre nicht die eigentliche natürliche Schnittstelle einfach die gesprochene Sprache?

U: Darüber kann man streiten. Natürlich haben Computerspezialisten unbewußt Angst davor, ihren Status zu verlieren. Wenn ein Systemprogrammierer sich schämt, mit Bildchen umzugehen, ist das Dummheit und Einbildung. Wenn er es lästig findet, weil er die ihm vertrauten Befehle auf der Tastatur schneller eingeben könnte, so ist das eine andere Sache. Wenn aber eine visuelle Benutzeroberfläche wirklich gut gestaltet ist, und wenn der Computer dafür leistungsfähig genug ist, dann ist ein geübter Benutzer genauso schnell damit wie ein Fachmann, der seine Befehle ja auch lange geübt hat. Was glauben Sie, wie kann man ein Autorennen gewinnen – indem man am Lenkrad dreht und mit dem Fuß Gas und Bremse bedient, oder indem man auf einer Tastatur eintippt: »15 Grad rechts, 12 Prozent mehr Gas – Widerruf, 30 Prozent bremsen, 10 Grad links...«?

Was die Sprache betrifft – sicher ist die Sprache viel mehr noch etwas spezifisch Menschliches als bildlich wahrnehmbare Gestalten. Tiere können auch optisch Gestalten wahrnehmen – Sprache ist etwas viel Abstrakteres und Höheres. Daher haben wir damit ja auch noch so große Schwierigkeiten in der Computerwissenschaft. Aber die bildhaften Gestalten sind auch viel urtümlicher – sie sind uns viel angeborener als sprachliche Ausdrucksformen, die jeder von uns in seinem Leben erst bewußt lernen muß. Vielleicht kommt das Bild vor der Sprache (die Sprache benutzt ja auch »Bilder im übertragenen Sinne«), jedenfalls können wir viel leichter Computer durch Bilder für jedermann benutzbar machen als durch gesprochene Sprache.

M: Aber führt das nicht auch wirklich zur Vernichtung von Kompetenz und dem Bedarf von fachlich qualifizierten Arbeitsplätzen? Wird nicht die Qualifikation einer Sekretärin wertlos, wenn jeder ihre Arbeit einen Computer tun lassen kann?

U: Ja und nein. Sehen Sie, es gibt nützliche Kompetenz und unnütze, solche, die nur Hokuspokus ist. In einer stark religiös geprägten Gesellschaft könnte man sich vorstellen, daß Computerkompetenz von Hohen Priestern als Geheimwissen zelebriert wird, weil es ihnen Macht über die Unwissenden verleiht.

In einer modernen, aufgeklärten Gesellschaft ist für den Gebrauch von Computern ebenso wenig Kompetenz notwendig, wie für das Aufdrehen eines Wasserhahns. Aber jemand muß den Wasserhahn konstruiert und montiert haben...

A.Das machen dann die Roboter...

U: Lassen Sie mich noch fortfahren. Natürlich werden Qualifikationen überflüssig, die früher gebraucht wurden. In einem mittleren Betrieb könnte der EDV-Leiter ebenso überflüssig werden, wie der Heizer auf der Elektrolok, aber dafür werden Software-Ingenieure und Ergonomen gebraucht, die alle diese famosen visuellen Benutzeroberflächen programmieren. Die sind nämlich verdammt kompliziert.

M: Wahrscheinlich aber kein Arbeitsplatz für den entlassenen EDV-Leiter?

U: Für viele nicht, für einige, die bereit sind, umzuschulen, Neues zu lernen, schon.

M: Nun wachsen unsere kleinen Kinder mit diesen Computer-Spiel-Umgebungen auf, wie früher mit Bauklötzen und Puppen. Wird das nicht einen tiefen Einfluß auf die Zukunft unserer Kultur haben?

U: Computer bringen eine neue Dimension in unsere Kultur. Der Umgang mit dem Computer ist eine neue Kulturtechnik. Bisher lernten wir Lesen, Schreiben und Rechnen, jetzt lernen wir auch noch Computern. Ein Freund sagte mir, während sich beim üblichen Rollenspiel der kleinen Mädchen Puppe und Kind so verhalten wie Kind und Mutter, sei es bei Computer und Kind eher wie Computer und Vater.

Oder – wenn das Kind zeichnet und dabei einen Computer benutzt, fühlt es sich in einer erwachsenen Rolle, identifiziert sich mit der Mutter, die mit dem Computer die Haushaltsabrechnung macht. Das heißt, das Kind wird kompetent und lernt, mit modernen Instrumenten des Wissens umzugehen, und seine Fähigkeiten selbst zu verstärken und zu erweitern.

A: Werden die Bildchen nicht eher den gerade in hochentwickelten Ländern wie den USA beobachteten modernen Analphabetismus fördern?

U: Ich kann Ihnen da nicht so ohne weiteres widersprechen. Die kleine fünfjährige Tochter Nuria unserer spanischen Bekannten, die fließend Deutsch und Spanisch spricht, lernte spielend – im Sinne des Wortes – mit dem Zeichenprogramm GEMDRAW umzugehen, sie konnte Figuren anklicken und die verschiedenen Zeichenmittel verwenden. Aber weil sie noch nicht lesen kann, fand sie die Pull-Down-Menüs, über denen gedruckte Wörter wie FONT, PATTERN und so weiter standen, schwer wieder.

A: Finden Sie das nicht beängstigend?

U: Ich weiß es nicht. Ich glaube, daß ein Kind ebenso mit Begeisterung sich selbst Lesen und Schreiben beibringen wird, wenn es einen Computer hat, der malen und sprechen kann, auf dem es auf Worte deuten und über dessen Tastatur es diese sonderbaren Zeichen eintippen kann. Aber da ist noch ein anderer Aspekt.

Während wir, nachdem Nuria sich sattgespielt hatte, über die verschiedenen nationalen Zeichensätze des Computers technisch diskutierten, übte sie mit ihrer Mutter mit herkömmlichem Papier und Kugelschreiber Schiffe versenken in einem Raster, das mit Buchstaben und Zahlen versehen war, und das Alphabet und das Schreiben von Wörtern wie CARMEN und SABINE; die Mutter vermerkte in einer Strichliste viel mehr »Buenos« als »Errores«.

U: Ich glaube, daß Kinder sehr wohl den Computer als eine Sache unter vielen in ihre Welt einbeziehen können. Wenn die Erwachsenen wieder mehr Zeit haben, ihnen auch andere Geheimnisse beizubringen, wird ein Kind den Computer nicht als Fetisch betrachten, sondern als ein Ding, das ihm als Menschen, der mit anderen Menschen agiert, zur Verfügung steht. Es ist dann nichts Erschreckendes, daß ein Ding auch intelligent sein kann.

A: Ist das nicht zu rosig gedacht?

U: Vielleicht. Aber das hängt davon ab, was wir mit unserer Gesellschaft anstellen. Computer sind eine Verführung – in demselben Sinne, wie Spielautomaten eine sind. Es ist aber nicht die Computerhardware, die unsere Kultur verändert, sondern die in Computerprogrammen von Menschen niedergelegten Gedanken. Was unsere Kultur verändert, sind also menschliche Ideen. Die können freilich auch unmenschlich verwendet werden – mit oder ohne Computer.

Aber sehen Sie – ich teile durchaus Ihre Besorgnis, daß die Computerkultur den Analphabetismus fördert. Wenn wir überall intelligente Maschinen um uns haben, nur auf das Bild eines Hamburgers deuten müssen, wenn wir hungrig sind und bezahlen, indem wir unsere Computerkreditkarte in einen Schlitz stecken, dann brauchen wir nicht mehr rechnen zu können, ja, nicht einmal mehr zu reden. Nur – selbst wenn diese Gefahr wirklich besteht, hat es keinen Zweck, darüber in Jammern und Wehklagen auszubrechen oder Zigarettenautomaten zu zertrümmern. Dies ist nur die konsequente Fortsetzung der Medienkultur. Menschen fahren Auto – dadurch ist unsere Vorstellung über die Entfernung zwischen München und Nürnberg eine andere als früher. Durch das Flugzeug sind selbst Kontinente nur Tagesreisen weit weg. Wir haben Telefon, Radio und Fernsehen – dadurch wissen wir mehr über die Welt, aber wir sind auch immer weniger erschüttert, wenn wir erfahren, daß irgendwo Menschen durch ein Erdbeben umgekommen sind. Nun verwachsen Telefon, Radio, Video und Computer immer mehr zu einer einheitlichen Technik – dem Mediencomputer. Wir müssen lernen damit umzugehen, aber aufhalten können wir es nicht.

Wir sollten unseren Kindern Wärme und Sicherheit vermitteln, ihnen zeigen, daß wir Zeit für sie haben – und ebenso Geduld und Respekt für die Selbständigkeit, mit der sie mit den Mediencomputern umgehen, selbständiger vielleicht als wir, die wir nicht damit aufgewachsen sind.

M: »Buenos« – das sind die »richtigen Lösungen« auf spanisch, nicht wahr? Wir haben da doch ein Textprogramm...

U: Sie meinen First Word, vielmehr 1ST Word, wie es sich schreibt. Auf dem – (Ungruen machte eine Geste mit den Fingern, die das Atari-Zeichen andeuten sollte. Er spielte damit darauf an, daß man dieses Zeichen mit dem vorher erwähnten Editor in den Text einfügen kann).

M: Ja. Wir haben über Analphabetismus gesprochen, den viele Leute befürchten, aber wenn ich an diese fantastischen Programme denke – da ergeben sich doch früher unvorstellbare Möglichkeiten für die Produktion von schriftlichen Dokumenten durch Privatpersonen.

U: Es ist sicher nicht übertrieben, das Erscheinen von Mediencomputern ähnlich einzuordnen wie die Erfindung der Buchdruckerkunst. Noch vor ein paar Jahren hatte ein gewöhnlicher Bürger, um sich schriftlich zu äußern, nur Bleistift, Füller oder Kugelschreiber, höchstens eine Schreibmaschine, zur Verfügung. Ein moderner Mediencomputer kostet nicht viel mehr als eine gute Schreibmaschine – er kann aber nicht nur Texte setzen, formatieren und korrigieren, er kann auch verschiedenartige Hervorhebungen, auf manchen Systemen auch schon unterschiedlicher Schriftgrößen und Schriftformen, sogenannte Fonts, hervorbringen. Außerdem kann man mit ihm zeichnen (auch wenn man kein Talent hat), Berechnungen anstellen und – mit den meisten dieser Systeme – Text und Grafiken kombinieren. Man kann also mit so einem Computer im Prinzip selbst eine Zeitschrift oder ein illustriertes Buch herstellen

Hand anlegen ... mit der Maus

M: Sie haben vorhin etwas in dem Sinne gesagt, daß diese modernen Benutzeroberflächen magisches Denken fördern oder verlangen. Bisher hat man doch immer gemeint, Computer förderten einseitig das rationale Denken.

U: Es gibt eine spielerische Magie. Wenn Sie einen fantastischen Roman lesen, einem Zauberer zusehen oder in einen Zeichentrickfilm gehen, dann spielen Sie doch mit der Unterscheidung zwischen vorgestellter Wirklichkeit und Realität draußen auf der Straße. Computer machen das Vorstellbare real. Um aber etwas zu erfinden, was es sonst nicht gibt, das man aber mit dem

Computer verwirklichen kann, braucht man Fantasie. Und man braucht Fantasie, um es anzunehmen – oder sagen wir, geistige Flexibilität. Deswegen meine ich auch nicht, daß uns die modernen Computeroberflächen in erster Linie verdummen – sie können auch eine Einladung sein, an das Erstaunliche zu glauben, daran, daß man einen Strich nehmen, verschieben, verbiegen und vervielfachen kann. Daran, daß man ein Auto durch eine einfache Handbewegung in einen flachen, schnittigen Rennwagen oder eine gemütlich-altertümliche Kutsche verzerren kann.

Der Mensch hat ein natürliches Bedürfnis nach Magie – das sehen Sie an jeder Boulevardzeitung, in der Horoskope abgedruckt werden. Vielleicht könnte der Computer dieses menschliche Bedürfnis in sinnvoller Weise ausfüllen – weil wir nämlich beim Computer immer zwischen der spielerischen oder künstlichen Natur, der gewollten Magie und der Realität unterscheiden. Aber beim Computer geht dadurch, daß wir wissen: es ist nur eine menschliche Erfindung, der Reiz nicht verloren. Das liegt daran, weil die Idee im Computer ein Eigenleben führt, weil man sie anfassen kann, weil sie Widerstand leistet – oder aktive Hilfe; sie ist lebendig.

Wenn ich noch einmal auf die Textprogramme auf diesen neuartigen Computern zurückkommen darf: Der (wieder das Atari-Zeichen) hat einen internationalen Zeichensatz. Geliefert wird er in einer Grundversion – zum Beispiel amerikanisch oder deutsch oder spanisch. Die jeweiligen Zeichen – also im Deutschen ä, ö, ü, Ä, Ö, Ü und ß, kann man über die Tastatur eingeben. Der Computer kann aber immer alle Zeichen, auch ein spanisches ñ, Ñ oder die merkwürdigen Satzzeichen ¿ (steht im Spanischen zu Beginn eines Fragesatzes) und ¡ (zu Beginn eines Ausrufs). Gleichzeitig kann er französische Zeichen wie ç, Ç, é und so weiter. Dazu kommen dann die wichtigsten griechischen Buchstaben, mathematische und logische Sonderzeichen. Alle diese Zeichen, vor allem ihre gleichzeitige Benutzung, stellten bisher die Hersteller von Schreibmaschinen oder Computern vor schier unüberwindliche Probleme. Das Textprogramm 1ST Word löst dies spielend – auf dem Bildschirm wird eine Tabelle aller verfügbaren Sonderzeichen eingeblendet, mit Hilfe der Maus »drückt« man das gewünschte Zeichen – wie auf einer imaginären Tastatur mit 256 Tasten – und das Zeichen erscheint im Text.

A: Nochmal: Können diese neuen Computeroberflächen nicht auch zu einem Verlust an Realitätssinn führen – so wie bei den Kindern, die glauben, daß Milch in der Fabrik hergestellt wird, weil sie noch nie eine Kuh gesehen haben. Konkret gefragt: Ein Kind, das weiß, daß man Texte beliebig editieren und Fehler verbessern kann, wird das nicht jedes Gefühl für das Dokumentarische verlieren? Wenn es eine Schularbeit mit Fehlern geschrieben hat, wird es nicht auch nachträglich seine Fehler korrigieren wollen, um ein Wissen vorzutäuschen, das es nicht hat?

U: Dazu muß es aber zwischen seinen Fehlern und dem richtigen Vorbild unterscheiden – und es ist die Frage, was uns wichtiger ist: Die Dokumentation eines Fehlers, oder die Fähigkeit, Fehler zu korrigieren, um Richtigkeit herzustellen. Schließlich sind Fehler zwar eine menschliche Schwäche, die Fähigkeit, sie richtigzustellen, aber eine menschliche Stärke.

A: Der nächste logische Schritt wäre, daß der Schüler einen Spelling-Checker (Programm, das die Rechtschreibung korrigiert) über seinen Aufsatz laufen läßt. Aber in der Realität werden Fehler eben nicht von selbst geheilt, sondern sie können verhängnisvolle Auswirkungen haben, ehe man sie erkennt. Muß man nicht lernen, seine Fehler zuzugeben, aus ihnen zu lernen, indem man sie aufzeichnet? Wenn man Unliebsames in einem gemalten Papierkorb verschwinden lassen kann, wird man da nicht auch glauben, ein Unrecht, das einem anderen angetan wurde, kann einfach vergessen, eine Grausamkeit wegeditiert werden?

U: Man kann ja im Computer auch die Veränderung speichern, also die Geschichte der Fehlerkorrektur. Das ist ebenfalls Bestandteil unserer Realitätserfahrung – wir korrigieren Fehler, aber deswegen haben wir sie doch einmal gemacht. Aber wahrscheinlich wollen Sie auf etwas ganz anderes hinaus – natürlich ist zum Beispiel eine Geschichte, die nur noch im Computer gespeichert wird, viel retuschierbarer als eine, die auf Papier gedruckt wurde.

A: Genau.

U: Wir müssen irgendwann begreifen, daß Ideen keine materiellen Güter sind – Programme sind ohne wesentliche Kosten reproduzierbar. Trotzdem können sie einen hohen Wert darstellen. Bestimmt wird die Erfahrung, wie Information mit dem Computer manipuliert werden kann, auch unsere Kritik schärfen. Wir wissen, daß das, was auf dem Bildschirm steht, deswegen noch lange nicht glaubhaft ist. Wir glauben, was in der Zeitung steht, wenn wir selbst keine drucken können, wir glauben, daß wahr ist, was man im Fernsehen sieht, solange wir nicht selbst mit der Videokamera gearbeitet haben.

Die Rolle der modernen Computeroberflächen liegt eben darin, daß das jeder erfahren kann – bisher konnten wird das, was der Computer errechnete, nur für eine höhere Wahrheit halten, weil nur Spezialisten verstanden, wie der Computer zu seinen Ergebnissen kommt. Wenn aber jeder Mensch ohne besondere Qualifikation erfährt, daß alles, was der Computer macht, vom Menschen beeinflußt wird – dann, so hoffe ich, wird er nie mehr einem Computer mehr glauben als seinem gesunden Menschenverstand.

M: Professor Ungruen, wir danken für dieses Gespräch.

(le)

Magische Grafikprogramme machen aus dem Töff-Töff einen Sportwagen

Bitte ein Bit

Assembler gilt als schwer erlernbar. Trotzdem, es ist und bleibt der Traum jedes Programmierers, diese Sprache zu beherrschen, denn sie ist an Geschwindigkeit und Kompaktheit kaum zu überbieten.

Wie aber ist nun Assembler in die Vielzahl der existierenden Programmiersprachen einzuordnen? Im Grunde verstehen Computer (und da machen die 68000er keine Ausnahme) nur binäre Kombinationen, das heißt, Kombinationen aus »0« und »1«. Das war schon in der grauen Vorzeit der Computertechnik nicht anders. Die kleinste Dateneinheit (0 oder 1) bekam damals den Namen Bit. Später faßte man mehrere dieser Bits zusammen und konnte die Programme etwas schneller eingeben. Acht Bits zusammen ergeben so ein Byte, mit dem man heute meist arbeitet.

Als dann die einzelnen Befehle, die ein Prozessor verstehen kann, noch durch verständliche Kürzel gekennzeichnet wurden (zum Beispiel ADD für addieren), war der Schritt zum Assembler getan. Assembler ist also nicht, wie des öfteren behauptet, mit der Maschinensprache identisch, sondern eher eine Klartextfassung von ihr. Deshalb wird die Assemblerprogrammierung des 68000 selbst dem Anfänger keine großen Probleme bereiten, da der Befehlssatz des 68000 noch dazu verhältnismäßig klein und modular aufgebaut ist. Trotzdem können auch komplexe Probleme in Assembler einfach programmiert werden.

Großzügiges Innenleben

Wie Sie sicherlich bereits wissen, zählt der 68000 zu den 16-Bit-Prozessoren mit interner 32-Bit-Struktur. Das bedeutet, daß innerhalb des Prozessors Zahlen bis zu 32 Bit Breite verarbeitet werden können (dezimal sind das Zahlen bis zu 4.294.967.295). Nach außen werden aber immer nur 16 Bit gleichzeitig übertragen. Seinen 8 Daten- und 7 Adreßregister rüsten den 68000 bestens für den 32-Bit-Betrieb aus. Adreß- und Datenregister können die gleichen Daten enthalten, haben aber für den Prozessor unterschiedliche Aufgaben. Der 68000 kennt zwei verschiedene Betriebsmodi, den Usermodus und den Supervisormodus. Im Usermodus können Sie weder alle Befehle nutzen noch auf geschützte Speicherbereiche zugreifen. Der Supervisormodus kennt solche Beschränkungen nicht. Diese Unterscheidung aber unterstützt auf sehr einfache Weise einen Multiuserbetrieb. Der Anwender kann so nur auf bestimmte Bereiche zugreifen, soweit es der Betriebsmodus zuläßt, und somit keinen Schaden anrichten. Im Einplatzsystem (zum Beispiel Atari ST oder Amiga) werden auf diese Art und Weise die Systemvariablen und die Bereiche der Peripheriebausteine geschützt. Der 68000 besitzt für jeden Modus einen eigenen Stackpointer, um Komplikationen beim Wechseln des Modus zu umgehen. Der aktuelle Prozessorstatus kann im Statusregister gelesen und auch direkt verändert werden. Dieses 16 Bit große Register teilt sich wiederum in ein User- und ein Supervisor-Byte (siehe Bild). Das Userbyte, auch Condition Code Register (CCR) genannt, enthält Informationen darüber, ob sich bei Operationen eine Null oder eine negative Zahl, oder ein Überlauf oder ein Übertrag ergibt. Diese Informationen sind bitweise in Flags gespeichert. Es existieren mehrere Flags (Zero-, Negativ-, Carry-, Extend- und Overflow-Flag), die für verschiedene Anzeigen zuständig sind. Ist ein Ergebnis gleich Null oder negativ, wird das Zero- oder Negativ-Flag gesetzt. Sollte beim Addieren oder Schieben der höchsten Bits ein Übertrag entstanden sein, ist das Carry-Flag gesetzt und das Overflow-Flag zeigt an, daß der zulässige Zahlenbereich überschritten wurde. Das Extend-Flag verhält sich im allgemeinen wie das Carry-Flag, wird aber von einigen Befehlen, die das Carry-Flag setzen oder löschen (zum Beispiel CMP), nicht beeinflußt.

Das Supervisor-Byte enthält die Interruptmaske, die bestimmt, welche der sieben möglichen Interrupt-Ebenen zugelassen werden. Ein Interrupt ist eine Unterbrechung, die beispielsweise eine Tastaturabfrage zuläßt. Dem Supervisor-Byte des Statusregisters ist zu entnehmen, ob sich der Computer im Supervisor- oder im User-Modus befindet. Die letzte Information, die das Register enthält, ist das Tracebit. Bei einem Bitwert von »1« springt der Prozessor, nachdem ein Befehl vollständig abgearbeitet wurde, zu einem festen Vektor. Dadurch ist eine Einzelschrittabarbeitung von Assemblerprogrammen einfach per Software realisierbar.

Universell: der MOVE-Befehl

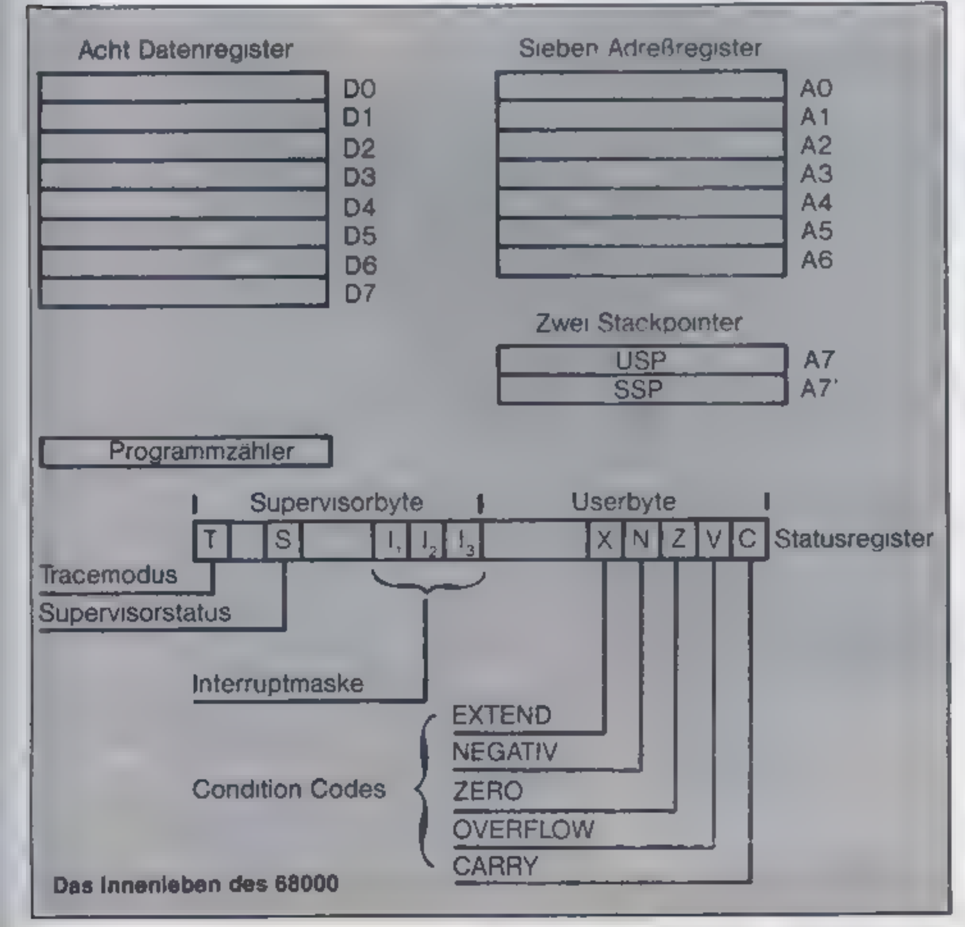
In den Assemblerprogrammen des 68000 spielt der MOVE-Befehl eine zentrale Rolle. Sämtliche Datentransfers werden mit diesem Befehl ausgeführt. Da MOVE alle Adressierungsarten (Tabelle 1) erlaubt, soll dieser Befehl genauer untersucht werden. Der einfachste Datentransfer ist der zwischen zwei Datenregistern. »MOVE D0,D1« kopiert den Inhalt des Datenregisters »0« in Datenregister »1«. Dabei kann festgelegt werden, ob eine Übertragung von Byte, Wort oder Langwortdaten stattfinden soll. Die korrekte Schreibweise wäre also »MOVE.B, MOVE.W« beziehungsweise »MOVE.L«, abhängig von der Größe. »MOVE.W #22222,D0« schreibt 22222 unmittelbar ins Datenregister D0 und ermöglicht so die Arbeit mit Konstanten. Ein wichtiger Datentransfer läuft natürlich zwischen Prozessor und Speicher ab. »MOVE.B 1000,D0« lädt zum Beispiel den Inhalt der Speicherstelle 1000 ins Datenregister 0, während »MOVE.W D0,1000« den Inhalt von D0 in die Speicherstellen 1000 und 1001 schreibt (ein Wort besteht jeweils aus zwei Byte). Bei Speicherzugriffen mit Worten und Langworten (ein Langwort besteht aus vier Byte) muß die Speicheradresse eine gerade Zahl sein, da der 68000 sonst einen Adreßfehler erkennt. Eine elegantere und flexiblere Art Speicherzugriffe zu programmieren, bietet die indirekte Adressierung. Steht in Adreßregister A0 zum Beispiel 1000, würde mit »MOVE.B (A0),D0« der Inhalt der Speicherstelle 1000 in D0 übertragen. Das Laden von A0 übernimmt der Befehl »MOVEA«, also zum Beispiel »MOVEA.L 1000,A0«. Würde man nun den Inhalt von A0 um eins erhöhen, könnte mit dem gleichen Befehl (»MOVE.B (A0),D0«) die nächste Speicherstelle, also 1001, ausgelesen werden. Hier wird nun auch der Unterschied zwischen Adreß- und Datenregistern deutlich. Mit Datenregistern ist keine indirekte Adressierung möglich, dafür sind sie aber mit (fast) allen Befehlen direkt ansprechbar. Bei Adreßregistern kann dies nur über Spezialbefehle (»MOVEA«) geschehen. Datenregister dienen hauptsächlich als Ausführungsspeicher für Rechenoperationen und als Laufvariable für Schleifen. Adreßregister finden hingegen im Normalfall als Zeiger für Datenspeicher eigener Programme Anwendung. Der 68000 bietet komfortable Möglichkeiten, die Adreßregister als Zeiger einzusetzen. Mit dem sogenannten Postin-

krement können Adreßregister immer nach Ausführung eines Befehls erhöht werden. »MOVE.B (A0)+,D0« würde, vorausgesetzt in A0 steht 1000, den Inhalt der Speicherstelle 1000 auslesen und A0 um eins erhöhen (es enthält nun 1001). Bei der Verwendung von Wort- oder Langwortdaten wird das Adreßregister um 2 beziehungsweise 4 Byte erhöht. Das folgende Programm, es handelt sich um eine Verschieberoutine, verdeutlicht die Möglichkeiten des Postinkrements.

```
MOVEA.L        1000,A0       Quelle
MOVEA.L        2000,A1         Ziel
LOOP MOVE.B (A0)+,(A1)+
CMP.L          #1500,A0     Fertig?
BNE            LOOP
```

Dieses Programm verschiebt den Speicher von 1000 bis 1499 nach 2000. Die ersten beiden Zeilen laden die Startadressen des Quell- und des Zielspeichers in die Register A0 und A1. In der dritten Zeile wird ein Byte von der Quelle zum Ziel transportiert, wobei gleichzeitig die Zeiger um eins erhöht werden. Dann folgt die Abfrage, ob alle Bytes übertragen wurden. Ist dies nicht der Fall, wird wieder zur dritten Zeile des Programms verzweigt.

Das Gegenstück zum Postinkrement ist das Predekrement. Dabei wird erst das Adreßregister je nach Datengröße heruntergezählt, und anschließend auf die Speicherstelle zugegriffen, die im Adreßregister steht. Die Befehlszeile »MOVE.W D0,-(A0)« zählt den Inhalt von A0 um zwei herunter (ein Wort entspricht zwei Byte), würde also in unserem Beispiel 1000 auf 999 reduzieren und legt den Wert aus D0 in der Adresse ab. Steht nun in einem Adreßregister die Basisadresse einer Tabelle und möchte man auf ein bestimmtes Element zugreifen, ohne die Basis zu verlieren, gibt es für solche oder ähnliche Fälle beim 68000 die Adreßdistanz. Der Befehl »MOVE.B 100(A0), D0« addiert 100 zu dem Inhalt von A0, bildet daraus eine neue Adresse und überträgt den Inhalt dieser Adresse nach D0. Steht also 1000 in A0, wird der Inhalt der Speicherstelle 1100 in D0 geladen. Als Distanz darf maximal eine 16 Bit breite Zahl stehen. Wem das nicht flexibel genug ist, kann seinem Befehl einen Index mit auf den Weg geben. Ein solcher Index ist immer in einem beliebigen Register enthalten, das in der Adressierung mit aufgeführt ist. Angenommen in A0 steht 1000 und in A1 steht 500, dann würde der Befehl »MOVE.B 100(A0,A1),D0« den Inhalt der Speicherstelle 1600 in D0 schreiben. Die Adreßdistanz beträgt hier maximal 8 Bit Breite. In den letzten beiden besprochenen Adressierungsarten kann auch der Programmzähler anstatt des Adreßregisters angegeben werden. Mit "MOVE.w 100 (PC),D0« zum Beispiel wird das Wort eingelesen, das 100 Byte hinter dem Befehl im Speicher steht. Programme, die immer auf diese Adressierungsarten zurückgreifen, sind an jeder Stelle im Speicher lauffähig, während andere Programme mit festen Speicheradressen arbeiten und somit an eine Stelle gebunden sind.



Das Innenleben des 68000

Arithmetisches Allerlei

Mit dem MOVE-Befehl alleine läßt sich allerdings kaum ein Programm schreiben. Eine der wichtigsten Arbeiten des Prozessors ist das Rechnen. Der 68000 besitzt eine ganze Menge Befehle, um arithmetische Aufgaben zu bewältigen. Der ADD-Befehl addiert zwei Zahlen, speichert das Ergebnis und legt besondere Informationen über das Ergebnis im Condition Code Register (CCR) ab. »ADD.W D0,1000« zählt den Inhalt der Speicherstelle 1000 und der Inhalt von D0 zusammen und speichert das Ergebnis in Adresse 1000. Steht bei 1000 der Wert 100 und in D0 11, ändert sich nach Ausführung des Befehls 1000 in den Wert 111, und alle Flags sind gelöscht. Der Befehl SUB hat dieselbe Syntax wie ADD, dient aber, wie der Name schon sagt, der Subtraktion zweier Zahlen. Mit den oben benutzten Werten schreibt der Befehl »SUB.W D0,1000« das Ergebnis 89 in Speicherstelle 1000.

Eine andere Syntax weisen die Befehle MUL und DIV auf. Das Ziel muß grundsätzlich ein Datenregister sein, die Operandengröße ist fixiert auf Wort beziehungsweise Langwort. Der MUL-Befehl dient der Multiplikation zweier 16-Bit-Zahlen zu einer 32-Bit-Zahl Dabei ist eine Unterscheidung, ob das Vorzeichen (das höchste Bit) beachtet werden soll oder nicht, möglich. MULS multipliziert mit, MULU multipliziert ohne Vorzeichen. »MULS 1000,D0« multipliziert die Inhalte von 1000 und D0 miteinander und legt das Ergebnis in D0 ab. Das Zero- und das Negativ-Flag werden je nach Ergebnis verändert, Carry und Overflow gelöscht; das Extended-Flag wird nicht beeinflußt. Der DIV-Befehl dividiert eine 32-Bit-Zahl in einem Datenregister durch eine 16-Bit-Zahl beliebiger Herkunft und legt zwei Ergebnisse in dem Datenregister ab. Dabei steht, genau wie bei der Multiplikation, die Vorzeichenbehandlung frei. Die unteren 16 Bit des Datenregisters beinhalten das ganzzahlige Resultat der Division, die oberen 16 Bit den Rest. Angenommen, in der Speicherstelle 1000 steht 500 und in D0 steht 16, dann legt der Befehl »DIVS 1000,D0« die Ergebnisse 31 (Quo-

tient) und 4 (Rest) in D0 ab. Um an die oberen 16 Bit zu kommen, gibt es zwei Möglichkeiten. Entweder man benutzt einen der Schiebebefehle oder aber den Befehl SWAP. SWAP hat die Aufgabe, die oberen 16 Bit eines Datenregisters mit den unteren 16 Bit zu vertauschen. Steht in D0 die Hexadezimalzahl $1111FFFF, findet man dort nach Ausführung des Befehls »SWAP D0« $FFFF1111. Die Schiebebefehle bieten eine weitere Möglichkeit, Werte gezielt zu verändern. Mit ihrer Hilfe kann man eine Zahl bitweise um eine oder mehrere Positionen nach links oder rechts schieben. Schiebt man zum Beispiel die Zahl 3 um eine Position nach rechts, ist das Ergebnis 6 (3 entspricht im binären System 011, einmal nach rechts geschoben ergibt das 110, das entspricht 6 dezimal). Nun kann aber noch bestimmt werden, was mit dem Bit geschieht, das auf der einen Seite hinausgeschoben wird, und was auf der anderen Seite angefügt wird (in dem Beispiel eine 0). Das Verändern einzelner Bits kann mit mehreren Befehlen geschehen. Mit AND, OR, EOR und NOT lassen sich die Boolschen Operationen bitweise durchführen. Das heißt, es werden bei zwei Zahlen die jeweils gleichwertigen Bits logisch miteinander verknüpft. Eine Ausnahme macht der NOT-Befehl, der nur einen Operanden benötigt. Er invertiert jedes Bit des Operanden. Die Befehle BCHG, BCLR, BSET und BTST dienen zum Invertieren, Löschen, Setzen und Testen einzelner Bits. Dabei ist jeweils die effektive Adresse und das gewünschte Bit anzugeben. Mit »BTST #5,D0« wird Bit 5 der Zahl in D0 abgefragt und das Ergebnis in das Zero-Flag geschrieben. Um dem Programmierer nun die Möglichkeit zu geben, Werte miteinander zu vergleichen und dem Ergebnis entsprechend im Programm zu verzweigen, besitzt der 68000, wie viele andere Prozessoren auch, den Compare-Befehl. Mit »CMP D0,D1« werden die Inhalte aus D0 und D1 miteinander verglichen und die Flags danach gesetzt. Der Vergleich ergibt sich dadurch, daß mit den beiden Zahlen eine Subtraktion durchgeführt wird, ohne das Ergebnis zu speichern. Nun kann aufgrund dieses Vorgangs mit einem der Branchbefehle eine Verzweigung erfolgen. BEQ (Branch on EQual) zum Beispiel verzweigt an die angegebene Stelle, wenn das Zero-Flag gesetzt ist. Ist die Bedingung nicht erfüllt, wird mit dem nächsten Befehl fortgefahren. Ein weiteres Beispiel für CMP und Branch gibt das obenstehende Programm. BNE bedeutet dabei Branch on Not Equal. Mit BMI (Branch on MInus) und BPL (Branch on PLus) kann das Negativ-Flag abgefragt werden. Für das Carry-Flag gibt es die Befehle BCC (Branch on Carry Clear) und BCS (Branch on Carry Set). BVC und BVS sind die entsprechenden Befehle für das Overflow-Flag. Der 68000 ist auch in der Lage, mehrere Flags abzufragen. BHI (Branch on Higher) zum Beispiel verzweigt nur, wenn weder Zero- noch Carry-Flag gesetzt sind, der BRA-Befehl (BRanch Always) immer. Leider ermöglichen die Branch-Befehle nur Sprünge bis maximal 32000 Byte vorwärts beziehungsweise rückwärts. Für längere Sprünge innerhalb des gesamten Adreßbereichs ist der Befehl JMP zuständig.

Unterprogramme erleichtern das Leben

Es kommt oft vor, daß man einige Programmstücke häufiger braucht und nicht jedesmal neu eingeben möchte. Hier bietet der 68000, wie viele andere Prozessoren auch, die Unterprogramme, die mit dem Befehl JSR (Jump to Sub-Routine) aufgerufen werden. Ein Unterprogramm wird grundsätzlich mit dem Befehl RTS (ReTurn from Subroutine) beendet. Der 68000 besitzt, wie man der Befehlsübersicht (Tabelle 2) entnehmen kann, eine Reihe weiterer Befehle. Würde man jeden einzelnen Befehl (es sind insgesamt 56), mit jeder für ihn möglichen Adressierungsart aufführen, hätte man es mit mehr als 1000 Befehlskombinationen zu tun. Deshalb kommen auch nur die wichtigsten Befehle zur Sprache. Die Übersicht (Tabelle 2) zeigt auch, wie leistungsfähig der Befehlssatz des 68000 ist. Es ist damit nicht schwer, Assemblerprogramme (relativ) einfach zu schreiben und sie auch mit einigem Geschick sehr übersichtlich zu gestalten. Hinzu kommt, daß der 68000 mit mehreren Befehlen die Unterprogrammstrukturen höherer Programmiersprachen, wie C oder Pascal, unterstützt. Der 68000 ist hiermit wohl einer der leistungsfähigsten Prozessoren unter den 16-Bittern, was hoffen läßt, daß sein Schattendasein hinter den Prozessoren 8088 und 8086 von Intel der Vergangenheit angehört. (Michael Bernards/aw)

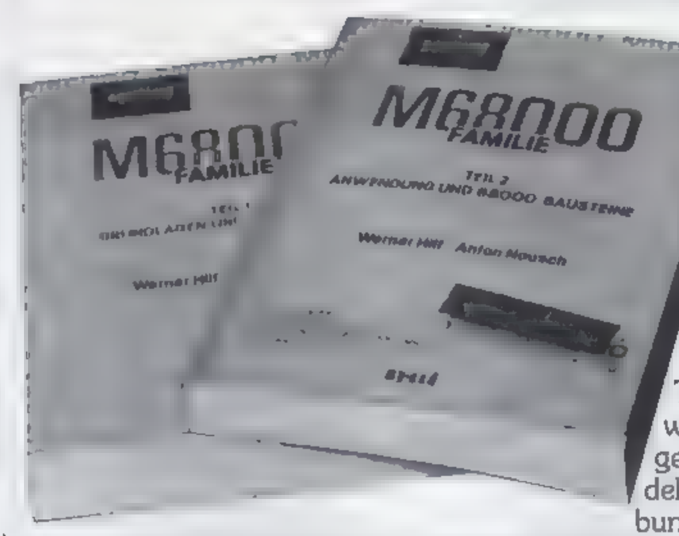
Dn	Datenregister direkt
An	Adreßregister direkt
(An)	Adreßregister indirekt
(An)+	Adreßregister indirekt mit Postinkrement
-(An)	Adreßregister indirekt mit Predekrement
d16(An)	Adreßregister indirekt mit Adreßdistanz
d8(An,Rx)	Adreßregister indirekt mit Adreßdistanz und Index
$XXXX	absolut kurz
$XXXXXXXX	absolut lang
d16(PC)	PC relativ mit Adreßdistanz
d8(PC,Rx)	PC relativ mit Adreßdistanz und Index
#,SR,CCR	Konstante und Statusregister
Erläuterungen der Kürzel:	
An	Adreßregister (0 < n < 7)
Dn	Datenregister (0 < n < 7
Rx	Register (0 < x < 15)
d8	8-Bit-Distanz
d16	16-Bit-Distanz
PC	Programmzähler

Tabelle 1. Adressierung des 68000

ABCD	Addition zweier BCD-Zahlen
AND	Logisches UND
ADD	Addition
ASL	Arithmetisch schieben links
ASR	Arithmetisch schieben rechts
Bcc	bedingte Verzweigung
BCHG	prüfe und invertiere Bit
BCLR	prüfe und lösche Bit
BRA	verzweige immer
BSET	prüfe und setze Bit
BSR	verzweige in Unterprogramm
BTST	prüfe ein Bit
CHK	prüfe Datenregister gegen Grenzen
CLR	Löschen eines Operanden
CMP	Vergleich zweier Operanden
DBcc	dekrementiere und verzweige bedingt
DIV	Division
EOR	exklusives Oder
EXG	vertausche Registerinhalte
EXT	vorzeichenrichtige Erweiterung
JMP	Sprung
JSR	springe in ein Unterprogramm
LEA	lade effektive Adresse
LINK	lege Stackbereich an
LSL	logisches Schieben links
LSR	logisches Schieben rechts
MOVE	übertrage Daten
MUL	Multiplikation
NBCD	Negation von BCD-Zahlen
NEG	Negation einer Zahl
NOP	keine Operation
NOT	logisches NOT
OR	logisches ODER
PEA	lege Adresse auf Stack
RESET	Rücksetzen der Peripherie
ROL	rotiere links
ROR	rotiere rechts
ROXL	rotiere links mit EXTEND
ROXR	rotiere rechts mit EXTEND
RTE	Rückkehr aus Exception
RTR	Rücksprung und Laden der Flags
RTS	Rückkehr aus Unterprogramm
SBCD	Subtraktion von BCD-Zahlen
Scc	setze bedingt ein Byte
STOP	halte Prozessor an
SUB	Subtraktion
SWAP	vertausche Registerhälften
TAS	prüfe und setze ein Bit
TRAP	springe in eine Exception
TRAPV	TRAP bei Überlauf
TST	Testen eines Operanden
UNLK	baue Stack ab

Tabelle 2. Die Befehle des 68000

M68000 FAMILIE



Dieses zweibändige Werk befaßt sich grundlegend mit der M68000-Familie, einer Familie von leistungsfähigen 16-/32-Bit-Prozessoren und der zugehörigen Peripherie.
Im ersten Teil wird die Architektur des M68000 ausführlich behandelt. Anschließend daran folgt die Beschreibung der Adressierungsarten und des Befehlssatzes des M68000.
Teil 2 bringt Programmierbeispiele, und es werden die bisher bekannten und zukünftigen Peripheriebausteine zum 68000 behandelt. Ein interessantes Kapitel bildet die Beschreibung der weiteren Prozessoren der M68000-Familie bis hin zum 68020, dem echten 32-Bit-Mikroprozessor Zum Schluß erhält der Leser noch eine vollständige Beschreibung eines Single Board-Systems auf der Basis des 68000. Diese Beschreibung umfaßt die Schaltpläne ebenso wie ein vollständiges Monitor source listing. Dadurch hat jeder Interessierte die Möglichkeit, sein eigenes 68000-System aufzubauen.
M68000-Familie, von Werner Hilf und Anton Nausch
Teil 1 – Grundlagen und Architektur, 550 Seiten, DM 79,–
Teil 2 – Anwendung und 68000-Bausteine, 350 Seiten, DM 69,–

Weitere te-wi-Bücher

UNIX™ – ANWENDERHANDBUCH
(R. Thomas, J. Yates)
Mit diesem Werk können sich Interessierte und Fortgeschrittene ein fundiertes Basiswissen über dieses Betriebssystem der Zukunft aneignen. Es ist so praxisnah geschrieben, daß man schon in kurzer Zeit mit UNIX am Computer arbeiten kann. (528 Seiten, Softcover) DM 79,–

UNIX™ – FÜHRER DURCH DAS SYSTEM
(Stanka/Lösch)
Das UNIX-System aus eigener, langjähriger Erfahrung der Autoren dargestellt. Systematisch. Urteilend. Anleitend. (264 Seiten, Softcover) DM 59,–

DIE C-SPRACHE (Z. Stanka/S. Lösch)
Dieses Buch dient der praktischen Konzipierung von C-Programmen. Logik und Struktur stehen im Mittelpunkt. (300 Seiten, Softcover) DM 59,–

ERFAHRUNGEN MIT GATE ARRAYS
(Andy Rappaport)
Bericht eines Anwenders über seine erstmalige Entwicklung eines kundenspezifischen Schaltkreises. (etwa 200 Seiten, Hardcover, viele Abbildungen) DM 66,–

EINFÜHRUNG IN DIE MIKROCOMPUTER-TECHNIK (A. Osborne)
Das schon legendäre Standardwerk des Autors wurde in dieser Ausgabe restlos überarbeitet. Es spiegelt den allerletzten Stand dieser bedeutenden Technologie wider. Geblieben ist aber die Osborne-typische Darstellung, in der auch vielleicht komplizierte Vorgänge bildhaft verständlich gemacht werden. Wer dieses Buch gelesen hat, kennt sich in der Welt der Mikrocomputer-Technik aus.
(480 Seiten, Hardcover) DM 66,–

te-wi Verlag GmbH
8000 München 40
Theo-Prosel-Weg 1

AmigaBasic – neuer Streich von Microsoft

Kaum ist der Amiga in Amerika auf dem Markt, kommt schon der Nachfolger seines ABasic. Durch ungewöhnliche Fähigkeiten hebt es sich deutlich von allen anderen ab. Was bringt es Neues?

Der Amiga nennt ein Basic sein eigen, das man kaum mit anderen Basicdialekten vergleichen kann. Ursprünglich sollte der Amiga mit ABasic von Metacomco ausgeliefert werden. Inzwischen entschied man sich bei Commodore aber anders: Das Microsoft-Basic hat den Vortritt. Ein Schritt, den Commodore nicht bereuen muß. Äußerst leistungsfähig und flexibel nutzt das AmigaBasic die Fähigkeiten des Commodore-Amiga voll aus.

Was als Allererstes auffällt: Man muß den Basic-Interpreter nicht erst laden, um Basic-Programme auszuführen. Man kann direkt aus der Workbench ein Basic-Programm starten. Der Interpreter wird vorher noch automatisch geladen und das geht übrigens recht flott. Eine Reihe von Demoprogrammen, die dem Basic bereits beiliegen, machen es noch attraktiver. Dabei handelt es sich unter anderem um einen Editor für Sprites und »Bobs« (= »Blitter Objects«, auch als Softwaresprites zu bezeichnen), ein Terminalprogramm, das mit 9600 Baud arbeitet (ein paar kurze Basic-Zeilen genügen dafür!), ein Musik-, ein Sprach- und verschiedene Grafikdemos.

AmigaBasic arbeitet völlig ohne Zeilennummern. Die hat es auch nicht mehr nötig, denn Befehle für strukturierte Programmierung machen dem Anwender das Arbeiten leichter und übersichtlicher. Wer nicht darauf verzichten mag, braucht aber nicht weiterhin auf Programmzeilennummern zu verzichten. Man kann einzelnen Zeilen aber auch Namen geben, ein »GOTO Name« genügt. Unterroutinen lassen sich mit Namen aufrufen, fast wie in Pascal. Strukturen wie WHILE/WEND- und REPEAT/UNTIL-Schleifen gestalten den Programmierstil übersichtlich. Als Nonplusultra kann man mit Hilfe des Screen-Editors Leerzeichen vor den Text einfügen. Manch einen beschleicht dabei das Gefühl, es mit Pascal-Programmen zu tun zu haben. Derselbe komfortable Screen-Editor, den man per Menüleiste oder Arbeitsfenster als List-Window aufruft, ermöglicht es, Basic-Programme einzugeben oder zu listen. Der Editor verfügt über Fähigkeiten, die man eigentlich nur von Textverarbeitungen, wie beispielsweise »MacWrite«, kennt. Die betreffenden Optionen können übrigens sowohl per Maus über Drop-Down-Menüs, als auch mittels Tastatur aufgerufen werden.

Störend wirkt einzig und allein das Zeitlupentempo, in dem das Scrolling erfolgt. Hier wären einige Verbesserungen angebracht. Das Arbeitsfenster erweist sich als relativ überflüssig. Da es keinen bildschirmorientierten Editor enthält, dient es im Grunde nur als Window zum Herumprobieren im Direkt-Modus. Wurde kein anderes Window vom Basic-Programm aus angesprochen, erfüllt das Arbeitswindow gleichzeitig auch den Zweck eines Output-Windows für Text und Grafik.

Multitasking

Das führt nun schon zum nächsten Punkt: Windows. Ein einfacher Befehl überantwortet dem Benutzer volle Kontrolle über Window-Darstellung und ihre Ansteuerung mit der Maus. Das sieht dann in etwa so aus:

```
WINDOW 1,"Windowname",(x,y)-
(x1,y1),Zahl
```

Die x- und y-Werte geben die Anfangskoordinaten und die Größe des Windows an. Diese letzte »Zahl« ist die Kontrolle darüber, wieviele und welche Funktionen dem Benutzer mit diesem ersten Window gestattet sind. Ein Bit davon ist zuständig für das Symbol, das in der rechten unteren Ecke des Windows erscheint. Ist es gesetzt, erlaubt die Maus eine Veränderung des Windows, andernfalls nicht. Die anderen Bits verursachen – falls sie gesetzt sind – das Verschieben eines Windows, das Ausklicken des Windows (das Viereck mit dem kleinen Punkt links oben am Window), oder das Plazieren eines Windows in den Vorder- oder Hintergrund (Zeichen rechts am Window). Nach denselben Regeln funktionieren Window 2, Window 3 etc. Innerhalb des Basic-Programms wird nun mit »WINDOW OUTPUT Nummer« festgelegt, welche Ausgabe in welches Window soll. So ist es möglich, verschiedene Applikationen in mehreren Windows gleichzeitig laufen zu lassen. Als echtes Multitasking ist das allerdings nicht zu bezeichnen, da man nicht verschiedene Basic-Programme von Disk laden und gleichzeitig laufen lassen kann. Vielmehr sind innerhalb eines Listings die Tasks als spezielle Unterroutinen zu definieren. Die Auswahl, auf welches Window die Ausgabe erfolgen soll, liegt dann in jeder Unterroutine fest. Dies gaukelt ein scheinbares Multitasking vor, die verschiedenen Inhalte müssen allerdings alle innerhalb eines Programmlistings auftreten. Mehrere Demoprogramme in verschiedenen Grafikwindows machen sich beispielsweise sehr hübsch. Die Grafikbefehle nehmen dann innerhalb eines »Tasks« die linke obere Ecke des jeweiligen Windows als Koordinate 0,0 an. Dies erleichtert das Arbeiten mit den Windows wesentlich. Es erspart nämlich sämtliche Berechnungen, wo zum Beispiel auf dem Bildschirm eine Grafik beginnen muß, um nicht etwa hinter einem Window zu verschwinden.

AmigaBasic unterstützt die Grafik des Systems. Dazu hält es Linien-, Box-, Circle-, Punktbefehle und noch so einiges mehr bereit. Um spezielle Dinge wie den »Hold & Modify«-Modus zu erreichen, helfen allerdings keine normalen Grafikbefehle mehr, sondern nur noch diverse Tricks. Dazu später mehr.

SOUND- und WAVE-Befehle sprechen die Musikfähigkeiten des Amiga an, natürlich auch in Stereo. Ein eindrucksvolles Demoprogramm demonstriert dem Anfänger, wie's geht! Einen besonderen Clou ließ sich Commodore mit seinen Sprachbefehlen einfallen. Dazu gibt man einfach »SAY A$, Optionen« ein; der String muß in Phonemcodes geschrieben sein. Die TRANSLATE-Funktion spricht die englische Sprache an. »SAY TRANSLATE$(A$), Optionen« veranlaßt den Computer also direkt zum Sprechen englischer Sätze. Die angefügten Optionen veranlassen den Computer, sich laut oder leise, in weiblicher oder männlicher Tonlage zu artikulieren. Das einzige Handicap: Zur Sprachausgabe ist die Workbench-Diskette notwendig, von der die entsprechenden »Translator« und »Narrator«-Routinen erst geladen werden müssen. Besitzer von nur einem Laufwerk fühlen sich dabei

manchmal mehr als Diskjockey, denn als Programmierer.

Die Stringverarbeitungsbefehle von AmigaBasic bieten sich geradezu für die Programmierung von Textverarbeitungen oder anderer Business-orientierter Software an. Auch Arithmetik-Fans gehen nicht leer aus. Durch einfache Definitionsbefehle kann man dann zum Beispiel nicht nur Zehner-Logarithmen berechnen, sondern auch umständliche Befehle wie »PRINT LOG 2of5« ausführen lassen. Der Aufwand mit »DEF FN« und ähnlichen Befehlen, die man von anderen Basic-Dialekten her kennt, erübrigen sich. Natürlich braucht man kaum noch erwähnen, daß Mausabfragen, Cursor- und Bildschirmansteuerung, Farbsetzung oder gar Timerabfragen berücksichtigt wurden. Sprites und Bobs sind kein Problem, entsprechende Befehle sorgen auch vom Basic aus für fließende (wenn bei den Bobs auch manchmal flimmernde) Animation.

Spitzenreiter

Eines macht dieses Basic nun allerdings zum absoluten Spitzenreiter : Es kann uneingeschränkt auf alle Betriebssystemroutinen zugreifen. Für den fortgeschrittenen Programmierer ist also ein C-Compiler nicht gleich ein Muß. Dieses Basic schon stellt ihm frei, zuerst das Betriebssystem eingehend zu erforschen. Das Amiga-Betriebssystem wurde in C geschrieben; die DOS-Routinen sind in Form von C-Funktionsbibliotheken organisiert, also mit Namen aufrufbar. Das AmigaBasic erlaubt es, mit Hilfe von DECLARE FUNCTION und CALL-Befehlen, alle diese Routinen mit entsprechendem Namen aufzurufen. Der Amiga kann also bis ins letzte Detail ausgenutzt werden und Dinge leisten, die Basic normalerweise nicht sieht. Beim Experimentieren mit den Betriebssystemroutinen störte nur, daß das Kernel-Manual aus dem Entwicklungssystem nicht gerade den allerneuesten Stand aufwies. Schade, denn so hätte man einiges aus dieser Supermaschine herauskitzeln können.

Der Gesamteindruck zum AmigaBasic ist sehr positiv. Das Arbeiten ohne Zeilennummern unterstützt das strukturierte Programmieren sehr. Durch den guten Screen-Editor macht das Programmieren viel Spaß. Dieses Basic hat mit seiner Leistungsfähigkeit und komfortablen Bedienung den sprichwörtlichen Vogel abgeschossen und wohl auch alle anderen Basic-Dialekte der weiter verbreiteten 16-Bit-Computer übertroffen.

(Manfred Kohlen/hb)

Grafikwunder ST-Basic

Das Basic des Atari ST braucht sich vor anderen Basic-Versionen nicht verstecken. Mit seinen Besonderheiten hat es einiges zu bieten.

Der Basic-Interpreter zum Atari ST wartet mit einigen Besonderheiten auf: große Programmcodelänge, Fenstertechnik, Fehlen von Prozeduren, Ähnlichkeit mit GW-Basic von Microsoft, sowie schnelle Ablaufgeschwindigkeit. Die Programmlänge von 138 KByte für den Basic-Interpreter des Atari ST kann man nur gewaltig nennen. Aber wo viel Speicher vorhanden ist, wird eben nicht mehr damit gespart. Allerdings hat er dafür einiges zu bieten.

Fensterputzen

Der Interpreter nutzt die Fenstertechnik von GEM. Nach dem Laden erscheinen vier Fenster auf dem Bildschirm: Das Editier-, Listing-, Kommando- und Ausgabefenster.

Links oben befindet sich das Listingfenster, das man am besten umgehend wieder löscht. Das Listing läßt sich darin nämlich nur darstellen, aber nicht editieren. Rechts daneben hat das Ausgabefenster seinen Platz, in derselben Größe wie das Listingfenster. Je nachdem wieviel Platz die Ausgabe erfordert, kann man es bis zum unteren Bildschirmrand verlängern. Größenveränderungen aller Fenster lassen sich auch während des Progammlaufes durchführen, ohne diesen zu unterbrechen; er verlangsamt sich nur wesentlich. Unter den beiden genannten Fenstern sitzt das Kommandofenster. Da die Kommandos längenmäßig nicht die gesamte Fensterbreite benötigen, ist es empfehlenswert, es zu verkleinern. Die halbe Bildschirmbreite erwies sich bei unserem Test als ausreichend.

Unter diesen dreien liegt etwas versteckt, das Editierfenster. Man kann es auf drei Arten aktivieren. Durch den Befehl »EDIT«, durch ein Anklicken oder durch das Kommando »START EDIT« in einem der Pull-Down-Menüs. Da nach dem Löschen des Listingfensters auf der linken Bildschirmhälfte Platz frei ist, macht es sich dort am besten.

Um ein Programm einzutippen, stehen das Kommando- oder das Editierfenster zur Wahl. Im ersteren spart man sich die Zeilennummer, da sie der Befehl »AUTO« auf Wunsch vorgibt. Leider läßt sich nur die gerade aktuelle Zeile bearbeiten. Den Cursor zum Zeilenanfang rückzuplazieren, ist nur mit »Backspace« möglich, und das löscht die Zeichen natürlich.

Mit kleinen Hindernissen

Verzichtet man auf die Zeilennummernvorgabe, eignet sich das Editierfenster besser. Es stellt einen guten Bildschirmeditor zur Verfügung, mit dem sich zufriedenstellend arbeiten läßt. Die bearbeitete Zeile hebt er hell hervor. Die Betätigung der Return-Taste verlagert sie in den Programmspeicher. Ist die Eingabe falsch, unterläßt man einfach den Druck auf die Returntaste. Die Funktionstaste 9 holt den Inhalt des Programmspeichers in den Editor, und die falsche Eingabe ist verschwunden.

Leider fehlt der Insert-Modus. Möchte man in eine Zeile etwas einfügen, muß man für jedes einzufügende Zeichen einmal die F1-Taste drücken. »F2« löscht das Zeichen rechts vom Cursor. »F3« stellt eine ganze Zeile zur Verfügung, mit »F4« läßt sich eine Zeile löschen. Für einen Editor ist etwas bedauerlich, daß eine Zeile, die nicht auf dem Bildschirm zu sehen ist, auch für den Cursor unerreichbar ist. Entweder man blättert seitenweise »F5« und »F6«, oder benutzt den vertikalen Rollbalken.

Großer Schlüsselbund

Um den eingegebenen Programmtext schnell zwischenzuspeichern, steht die Tastenfunktion »F7« zur Verfügung. Sie schreibt den Text in eine Datei namens BASIC.BUF. Um den Inhalt dieses Puffers zu laden, drückt man einfach »F8«. Es ist angenehm, sich bei dieser Datensicherung nicht durch das eigentliche Ein/Ausgabemenü »wühlen« zu müssen.

Die Vielfalt der Funktionen dieses Interpreters ist reichlich: Mehr als 130 Schlüsselwörter stehen zum Abruf

bereit; schade nur die geringe freie Speicherkapazität. Läßt man sich mit dem Befehl »PRINT FRE(0)« den freien Speicher anzeigen, ergeben sich enttäuschende 10 KByte beim ST mit 512 KByte RAM. Um diesen Mangel etwas erträglicher zu machen, steht es frei, die Desktop-Accessoires auf der Systemdiskette zu löschen oder umzubenennen. Sie werden dann beim Systemstart nicht mitgeladen. Arbeitet man dann noch ohne den Grafik-Puffer des Interpreters, erhöht sich der freie Speicherplatz auf zirka 70 KByte. Damit läßt sich schon eher etwas anfangen.

Klar sein muß man sich allerdings, daß ohne Grafik-Puffer erstellte Grafiken nicht temporär gespeichert sind. Öffnet sich nach dem Programmlauf zum Beispiel das vorher geschlossene, oder vom Ausgabefenster überdeckte Kommando-Fenster, dann löscht es dadurch diesen Teil der Grafik.

Über die Befehle »GEMSYS« und »VDISYS« lassen sich auch GEM- und VDI-Routinen ansprechen. Damit können auch in Basic Fensterfunktionen programmiert werden. Leider gibt das Handbuch darüber nur unzureichend Auskunft.

Flitzer

Die Ablaufgeschwindigkeit kann sich sehen lassen. Beeindruckend schnell zeichnet der Atari ST Grafiken auf dem Bildschirm. Effiziente Befehle unterstützen und vereinfachen die Grafikprogrammierung.

Tauchen im Programm Fehler auf – leider ist Basic dafür prädestiniert – helfen »Debug«-Befehle, ihnen auf die Spur zu kommen. »Entlausen« ist vom englischen Wort »Debuggen« abgeleitet, was dasselbe bedeutet. Um sich die gerade bearbeitete Zeile im Kommandofenster ausgeben zu lassen, gibt es die »Tron-Funktion«. Die komplette Zeile zeigt »Trace« an. In vielen Fällen genügt es auch, nur den Wert einer Variablen zu wissen, die gerade bearbeitet wird. »Follow« bedeutet »Folge« und mit diesem Befehl bleibt man einer Variablen auf der Spur.

Das 121 Seiten starke Handbuch aus ungebundenen und kopierten Blättern liegt zur Zeit nur in englischer Form vor. Es erklärt jeden Befehl in alphabetischer Reihenfolge mit wenigen Worten, meist anhand eines kurzen Beispieles. Die eigentlichen Hinweise auf den Gebrauch diese Interpreters beschränken sich auf nur klägliche 13 Seiten. Der Rest des Handbuches, als Anhang bezeichnet, ist auch in diesem Stil abgefaßt. Er verschlechtert leider den ansonsten guten Eindruck des Basic für den Atari ST beträchtlich.

(hb)

Exotisch in jeder Beziehung

Mit seinem Erscheinen setzte der Sinclair QL neue Maßstäbe auf dem Sektor der Heimcomputer. Dies gilt nicht zuletzt für das Super-Basic, das sich zusammen mit dem Betriebssystem QDOS im 48 KByte ROM befindet.

Ein sehr bedeutsamer Unterschied zu herkömmlichen Basic-Dialekten ist die Anwendung von Prozeduren. Dadurch kann man in Super-Basic strukturierte Programme schreiben, die vollkommen ohne »GOTO«- und »GOSUB«-Befehle auskommen. Da Übergabeparameter, globale und lokale Variablen zulässig sind, sehen Super-Basic-Programme oftmals Pascal-Programmen täuschend ähnlich, zumal auch Einrückungen am Zeilenanfang einer übersichtlichen Gliederung dienen können.

Da es Super-Basic aber nicht erforderlich macht, die verschiedenen Variablentypen vorab festzulegen und der Basic-Interpreter versucht, alle Eingaben sinnvoll auszulegen, erleichtert es gerade dem Neuling den Einstieg in die Programmierung ungemein. Freilich erzielt es diese Vorteile auf Kosten der Geschwindigkeit. Leider wird die Darstellung im Handbuch dem hohen Standard des Super-Basic nicht gerecht.

Ausbaufähig

Alle Super-Basic-Befehle sind in einer Tabelle gespeichert. Diese Tabellen kann der Anwender bei Bedarf erweitern, so daß sich jederzeit eigene Prozeduren oder Funktionen in Maschinensprache anfügen lassen. Verschiedene Toolkit-Programme für den QL machen sich diese Tatsache bereits zunutze.

Auch die Variablen eines Programmes verwaltet Super-Basic mit Hilfe einer Tabelle. Im eigentlichen Programm wird eine auftretende Variable durch einen Zeiger auf die entsprechenden Speicherzellen in der Tabelle ersetzt. In diesen Speicherzellen deutet ein Zeiger auf einen Eintrag in der Namensliste. Die Länge eines Variablennamens kann bis zu 255 Zeichen betragen; hierbei sind Bezeichner für Felder und Laufvariablen eingeschlossen. Ein zweiter Zeiger weist auf den aktuellen Wert der Variablen, der sich in einer Wertetabelle befindet. Dieser Aufbau spart enormen Speicherplatz, selbst wenn man Variablen mit langen Namen in Programmen häufig verwendet. Der Basic-Interpreter unterstützt in weitem Umfang die Bildschirmverwaltung. Der Programmierer kann Fenster definieren, eine Skalierung der Achsen vornehmen, und eine Turtlegrafik animiert zum Spielen mit Grafikprogrammen. Durch die Umrechnung beim Skalieren der Achsen arbeitet die Bildschirmeinstellung allerdings nicht pixelgenau. Es ist jedoch möglich, Befehle des Super-Basic durch eigene Maschinencode-Programme mit demselben Namen zu ersetzen, die es dann vorrangig verwendet. Bei Bedarf ist somit eine Anpassung an eigene Bedürfnisse nicht ausgeschlossen.

Zeilenweise editieren

Wer mit den kleinen Brüdern des Sinclair QL bereits gearbeitet hat, der kommt mit dem Editor schnell zurecht. Ein Außenseiter wird sich wundern. Das Super-Basic verfügt leider nicht über einen bildschirmorientierten Editor, sondern nur über einen Zeileneditor. Jede Zeile, die man editieren möchte, muß man in das Editierfeld kopieren. Erst dort lassen sich die gewünschten Veränderungen vornehmen. Anschließend kopiert man sie, durch einen Druck auf die Return-Taste, wieder in den Programmtext. Die Ursprungszeile wird dadurch überschrieben. Zeilen lassen sich damit sehr einfach kopieren.

Super-Basic ist eine Sprache, die in weitem Umfang die einfache Handhabung herkömmlicher Basic-Dialekte mit fortschrittlichen Sprachkonzepten vereint. Compiler für diesen Basic-Dialekt sind inzwischen auf dem Markt, die auch das Handicap der langsamen Arbeitsgeschwindigkeit erheblich mindern. Somit bleibt es lediglich der Selbstdisziplin des Programmierers überlassen, Problemlösungen zu finden, die an Eleganz und Effizienz Programmen in anderen höheren Programmiersprachen durchaus ebenbürtig, wenn nicht sogar überlegen, sind.

(Wolfgang Diekert/hb)

C + Pascal = Modula

Einsteiger sollten umsteigen. Da für die neue Generation von Computern ohnehin neue Software gekauft werden muß, sollte man sich genau überlegen, ob das bei der Mehrzahl der Käufer eher knappe Budget statt für einen Pascal- oder C-Compiler nicht besser in einem Modula-Entwicklungssystem angelegt ist.

Noch stoße ich bei vielen Kollegen und manchen Hackern mit meinem Enthusiasmus für Modula-2 auf einen zähen, irritierten Widerstand. »Das ist halt Dein Hobby« höre ich da, und manchmal komme ich mir vor, als billige man es meiner Narrenfreiheit zu, immer wieder über Modula zu schreiben.

Tatsächlich fragt sich so mancher geplagte Anwendungsprogrammierer: Modula – wer ist das? Noch so ein exotisches Gewächs wie Forth oder Comal oder Elan, eine Modeerscheinung, die in zwei Jahren vergessen sein wird?

Oder wenn nicht das, dann so ein von Geburt an ungeschlachter Saurier wie PL1 oder Ada?

Weder noch. Wenn Sie sich folgenden Programmabschnitt ansehen – in welcher Programmiersprache ist er wohl geschrieben?

```
PROCEDURE INCZaehler (VAR x0, x1:
CARDINAL);
(* zählt den aus zwei 16-Bit-Zahlen
bestehenden Zähler der
Eingabezeichen um 1 hoch *)
BEGIN
 IF x0 < MAX (CARDINAL)
    THEN INC (x0);
    ELSE INC (x1); x0 := 0;
 END (* IF *);
END INCZaehler
```

»Sieht aus wie Pascal ... kann es aber doch nicht sein«, würde der Kenner sofort sagen. Wenn Sie nur flüchtig mit Turbo-Pascal vertraut sind, werden Sie wahrscheinlich nicht auf Anhieb erkennen, wieso es kein Pascal sein kann. Daher will ich das einmal an diesem winzigen Beispiel Schritt für Schritt untersuchen (es stammt übrigens aus dem Programm »Superfilter« für den Atari ST unserer Zeitschrift »Computer persönlich«).

Daß es nicht Pascal und auch nicht C, sondern Modula ist, haben Sie gewiß schon erraten. Daß es nicht C sein kann, sieht auf den ersten Blick, wer schon einmal im Leben ein C-Programm gesehen hat.

Die erste Frage wirft der Datentyp der Parameter auf: CARDINAL. Ein Standard-Datentyp von Pascal ist das jedenfalls nicht. Könnte aber vielleicht ein im Programm selbst definierter Typ sein?

Nun sieht die Formulierung »MAX (CARDINAL)« in der IF-Bedingungsanweisung aber sehr nach einer Standard-Funktion aus, die wahrscheinlich bedeuten soll, daß x0 mit der größten darstellbaren Zahl vom Typ CARDINAL verglichen wird. Schaut man sich den (für jeden Pascal-Programmierer leicht verständlichen) Algorithmus zusammen mit dem Kommentar am Anfang der Prozedur an, so liegt die Annahme nahe, daß es sich um vorzeichenlose ganze Zahlen handelt. Tatsächlich ist CARDINAL ein numerischer Typ – es sind die positiven ganzen Zahlen einschließlich der Null.

Damit kann man auf einem 16-Bit-Rechner von 0 bis 65535 zählen (kein Vorzeichenbit). Pascals INTEGER geht nur bis 32767. »Darüber« tritt das auch von Basics POKE her bekannte Phänomen auf, daß Speicheradressen im oberen Bereich mit negativen Zahlen angesprochen werden (in Pascal »darf« man eigentlich gar nichts mit Speicheradressen anstellen, in Modula-2 sehr wohl).

Beim Blick auf das Ende dieser Zeile und den Anfang der nächsten muß aber der wahre Pascal-Kenner wie von der Tarantel gestochen aufspringen: Ein Semikolon (;) vor ELSE! Das ist keine korrekte Pascal-Syntax. Generationen von Pascal-Programmierern wurden vom Starrsinn des Compilers geplagt, daß jede Anweisung durch ein Semikolon abzuschließen ist, außer, es folgt ein ELSE! (Vor END durfte das Semikolon stehen oder entfallen, vor ELSE durfte keines stehen.)

Weitere Fragen stellt die folgende, von ELSE eingeleitete Zeile. Da hier zwei Anweisungen (INC (x1); x0:=0;) in einer Zeile stehen, beabsichtigt der Programmierer wahrscheinlich, daß sie beide im ELSE-Zweig ausgeführt werden sollen. Da fehlt dann doch aber ein BEGIN nach ELSE (von Pascal aus gesehen)! Das legt auch das END in der Zeile danach nahe – das zugehörige BEGIN fehlt dem kundigen Pascal-Auge. Schließlich kann auch »END INCZaehler;« (Wiederholung des Prozedurnamens ohne Kommentar-Klammern) nie und nimmer Pascal sein. Prozeduren werden in Pascal einfach durch »END;« abgeschlossen. In Pascal würde dieselbe Prozedur so aussehen:

```
PROCEDURE INCZaehler (VAR x0,
x1:INTEGER);
(* zählt den aus zwei 16-Bit-Zahlen
bestehenden Zähler der
Eingabezeichen um 1 hoch *)
BEGIN
IF x0 < MAXINT
  THEN x0:=SUCC(x0)
  ELSE BEGIN x1:=SUCC(x1); x0:=0;
       END (*ELSE *);
END (* INCZaehler *);
```

Das würde allerdings nicht korrekt funktionieren, falls es sich bei x0 und x1 tatsächlich um einen (aufgespaltenen) 32-Bit-Zähler handeln sollte. Das wäre dann so, als würde man im Dezimalsystem mit zwei Stellen bis 99 zählen wollen, an der Einer-Stelle käme aber nach »4« statt der 5 der Wert »-1«.

INTEGER-Zahlen (diesen Datentyp gibt es auch in Modula) verwenden das oberste der 16 Bit als Vorzeichen, damit ergibt sich ein Zahlenbereich von – 32768 bis +32767.

Noch ein Beispiel: Ich erhalte einen Anruf von einem kleinen Softwarehaus, das mit einem Grafikpaket (in UCSD-Pascal) arbeitet: Im Plottertreiber sei ein Fehler, da trete bei Schriften, die um andere Winkel als 0 oder 90 Grad gedreht werden, ein Floating-Point-Fehler auf. Bisher hat man keine Beschriftungen in nicht-rechten Winkeln geplottet, und nun stelle sich plötzlich heraus, daß andere Winkel nicht funktionieren.

Tatsächlich scheint der Fehler nicht im Plottertreiber zu liegen, sondern in meinem darüberliegenden geräteunabhängigen Grafiksystem – denn, so die mysteriöse Auskunft des Programmierers, manchmal funktioniert es, manchmal kommt einfach eine unsinnige Größenangabe für die Schriftgröße aus dem Grafiksystem an den Plottertreiber – scheinbar ohne jede feste Regel, nach einem Zufallsmuster. Die Eingabewerte aus dem von ihm geschriebenen Baustatik-Programm seien immer korrekt. Das erinnert mich an etwas – Fehler durch nicht initialisierte Variable können sich so verhalten und unter Umständen jahrelang unentdeckt ihr spontanes Unwesen treiben.

Schließlich fällt es mir beim Anblick des Codes von »DrawReal« wie Schuppen von den Augen, ich sehe spontan den (von mir programmierten) Fehler:

```
PROCEDURE DrawReal (x,y:REAL;
  r:REAL; vk,nk:INTEGER;
  sc,ri:REAL);
BEGIN
  ...
  IF selected THEN BEGIN
    CASE ROUND (ri / 90.0) OF
```

```
      0,2:        scale:=xpfact*sc;
      1,3:        scale:=ypfact*sc;
    END (* CASE *);
    ...(*weitere Anweisungen*)
  END (* IF *);
  ...
END (* DrawReal *);
```

Die Bedeutung der einleitenden Zeile mit der IF-Bedingung können wir ignorieren (in Modula würde das BEGIN natürlich wieder entfallen!). Interessant ist die CASE-Anweisung. »ri« ist die als Parameter übergebene Richtung (Winkel in Grad) des Schriftzugs, der geplottet werden soll. »sc« ist die gewünschte Schriftgröße, die der Prozedur als Parameter übergeben wird. »scale« ist eine lokale Variable dieser Prozedur, deren Wert an den Plottertreiber weitergegeben wird. Daß dies eine lokale Variable ist, und daß lokale Variable von Prozeduren in Pascal nicht initialisiert werden (wie etwa in Basic), darin liegt der Schlüssel des Problems!

Die von mir beabsichtigte Bedeutung dieser vier Zeilen Code ist folgende: Das Grafiksystem führt eine sogenannte Window-Viewport-Transformation aus, das heißt, die Zeichnung kann in unterschiedlichen Maßstäben wiedergegeben werden (verkleinert oder als in Vergrößerung gezoomter Ausschnitt). Dabei muß natürlich auch die Größe geplotteter Beschriftungen angepaßt werden. Das Maß dieser Transformation wird durch einen Faktor namens »Picture-Factor« ausgedrückt, allerdings getrennt für die X- und Y-Koordinaten (damit ist eine Verzerrung des Seitenverhältnisses möglich).

Dazu wird das Koordinatensystem in vier Quadranten unterteilt, die um 45 Grad nach links gedreht sind. Der erste Quadrant reicht also von 0 Grad (senkrecht nach oben) bis 45 Grad (X-Achse), der zweite von 45 bis 135 Grad (Y-Achse), der dritte von 135 bis 225 (X-Achse), der vierte von 225 bis 315 (X-Achse). Nun fehlen uns die Winkel von 315 bis 360 Grad, diese 45 Grad gehören natürlich zum ersten Quadranten. Na, haben Sie den Fehler schon erkannt?

Der Programmierer teilte mir mit, der Fehler träte beispielweise bei einer Schriftneigung von 315 Grad auf. Da 315 durch 90 gleich 3,5 ist, ergibt die Rundung hier den Wert 4 – und der fehlt in meiner CASE-Anweisung. Korrekt muß das CASE-Statement also lauten:

```
CASE ROUND (r1/90.0) OF
   0,2,4:      scale:=xpfact*sc;
   1,3:        scale:=ypfact*sc;
END (* CASE *);
```

In Modula könnte dieser Codeabschnitt beispielsweise folgendermaßen aussehen (dabei ist zwar der Fehler noch enthalten, doch würde er ein klareres Verhalten zur Folge haben):

```
IF selected THEN
  CASE ROUND (r1/90.0) OF
     0,2:        scale:=xpfact*sc;
     1,3:        scale:=ypfact*sc;
     ELSE scale:=0.0;
  END (* CASE *);
    ... (* weitere Anweisungen *)
END (* IF *);
```

Um in Pascal die gleiche Sicherheit zu erreichen, hätte dieser Codeabschnitt folgendermaßen lauten müssen:

```
IF selected THEN BEGIN
  IF (ROUND (r1/90.0) >0) AND
(ROUND (r1/90.0) < 3) THEN
    CASE ROUND (r1/90.0) OF
     0,2:        scale:=xpfact*sc;
     1,3:        scale:=ypfact*sc;
    END (* CASE *);
  ELSE scale:=0.0;
  ... (* weitere Anweisungen *)
END (* IF *);
```

In UCSD-Pascal wird, wenn in einer CASE-Anweisung der Wert des Ausdrucks nach CASE mit keiner der aufgeführten Konstanten übereinstimmt, einfach mit der nächsten Anweisung nach dem END (von CASE) fortgefahren. Das bedeutet, daß die lokale Variable »scale« niemals initialisiert wird, sie hat daher einen zufälligen Inhalt. Dieser zufällige Inhalt wird an den Plotter-Treiber ausgegeben, und bei einer Floating-Point-Variablen kann ein solcher Inhalt je nach »Vorgeschichte« des Speichers die unterschiedlichsten Auswirkungen haben.

Dabei ist dieses Verhalten von UCSD eine gutgemeinte »Verbesserung« der ursprünglichen Sprachdefinition. Da hieß es nämlich, wenn der Ausdruck einer CASE-Anweisung mit keiner Fallmarke übereinstimmt, ist die weitere Programmausführung »undefiniert«. Microsoft-Pascal auf dem IBM-PC nimmt das wörtlich – dabei erfolgt ein unkontrollierter Sprung an irgendeine Stelle im Programm – mit völlig mysteriösen Folgen.

Modula erweitert nun die Syntax der CASE-Anweisung. Es gibt genau zwei Möglichkeiten: Entweder, der ELSE-Zweig fehlt. Dann gibt das System, falls der Ausdruck bei CASE mit keiner Fallmarke übereinstimmt, eine Fehlermeldung aus und bricht das Programm ab. Man wird also sofort auf die Stelle aufmerksam, an welcher der Fehler aufgetreten ist – es kommen keine mysteriösen Folgefehler in anderen Programmabschnitten vor.

Oder, wenn der vorsichtige Programmierer den oben zitierten ELSE-Zweig geschrieben hätte, dann hätte er von seinem Kunden die Auskunft bekommen: Bei Winkeln zwischen 315 und 360 Grad ist die Schrifthöhe immer Null – woran liegt denn das?

Diese beiden kurzen Beispiele zeigen bereits sehr deutlich, daß der Kern von Modula-2 nichts anderes als ein syntaktisch modernisiertes und verbessertes Pascal darstellt – Modula ist eine Weiterentwicklung von Pascal. Doch hat es der schweizerische Schöpfer der beiden Sprachen, Niklaus Wirth, nicht bei kosmetischen Korrekturen bewenden lassen.

Fassen wir kurz zusammen, was Pascal bedeutet: Die strukturierte Formulierung von Algorithmen, wie sie aus Algol (ALGOrithmik Language) bekannt war, wurde weiterentwickelt und um die Möglichkeit, auch Datenobjekte zu strukturieren, ergänzt: Records, Arrays, Files, Records mit Arrays, Arrays mit Records als Komponenten, Files über alle Datentypen, Pointer zur Realisierung dynamisch verwalteter Datenobjekte und beliebiger Datenstrukturen wie Bäume und Graphen, benutzerdefinierte Datentypen und benutzerdefinierte Datenstrukturen.

Bei alledem war Pascal eine einfache Sprache – einfach sowohl für denjenigen, der sie lernte, als auch für denjenigen, der sie zu implementieren hatte – also für den Compilerbauer. Die Mängel von Pascal waren im wesentlichen: Keine Möglichkeit, Programme in voneinander unabhängige Bausteine zu zerlegen, die sich getrennt übersetzen lassen – insbesondere keine Möglichkeit, Bibliotheken mit wiederverwendbaren Routinen anzulegen. Keine Möglichkeiten, in Prozedurparametern »offene Arrays« (mit nicht festgelegter Anzahl von Elementen) zu verwenden (ist nach dem ISO-Standard jetzt möglich), wenige oder nur durch »Programmiertricks« erschließbare Möglichkeiten, die abstrakte Ebene der Hochsprache zu verlassen und direkt auf die Maschinenebene zuzugreifen, keine »sauberen« Möglichkeiten, Umwandlungen zwischen verschiedenen Datentypen durchzuführen (außer in ganz speziellen Fällen wie INTEGER/CHAR). Und das, obwohl alle Konstrukte von Pascal recht maschinennah sind, in dem Sinne, daß sie sich leicht und effizient auf konventionelle Maschinenstrukturen abbilden lassen. Pascal bot auch keine Möglichkeit, scheinbar oder tatsächlich parallel ablaufende Prozesse zu formulieren. Somit war diese Sprache in seiner usprünglichen Form überhaupt nicht zur Programmierung auf Systemebene geeignet.

Modula hat sich inzwischen als Bekenntnis bei den meisten Praktikern durchgesetzt, doch fehlte es bisher an überzeugenden und ausgereiften Implementierungen – was sich gegenwärtig mehr und mehr ändert. Leider gibt es aber auch hier Probleme mit der Einigkeit über die »Standard-Bibliotheken« und in einigen Details auch (noch) über die Definition der Sprache selbst.

So ist Modula viel mehr als ein abstraktes Konzept zu sehen, das in bestimmten Bereichen von der konkreten Implementierung unabhängig ist (die Syntax der Sprache und die Bedeutung von Standardprozeduren sollten überall exakt gleich sein), in anderen praktischen Bereichen aber abhängig von der jeweiligen Implementierung bleibt. Der Sprachentwurf versucht diese beiden Gebiete nach Möglichkeit klar zu trennen.

Ausdrücklich implementierungsabhängig sind die vom Pseudo-Modul SYSTEM importierten Konzepte wie der Typ WORD (oder der ihn häufig ergänzende Typ BYTE): Welche Wortbreite ein Prozessor hat, ob INTEGER- oder CARDINAL-Zahlen aus 16 oder 32 Bit bestehen, ob Adressen und Pointer 16 Bit umfassen (Z80) oder 20 (8088/86) oder vielleicht 32 Bit (68000/68020), kann man nicht in einer Sprachdefinition festlegen, weil sich die Computer weiterentwickeln.

Auch sogenannte Typ-Transfers (Uminterpretation von Typen ohne eigentliche Konversion) führen zu Unterschieden in den Implementierungen. Ein Typ-Transfer bewirkt, daß der Compiler die syntaktische Sperre zwischen verschiedenen Typen aufhebt, sofern sie physikalisch (auf Maschinenebene) gleich repräsentiert werden (zum Beispiel sind CARDINAL und INTEGER wahrscheinlich beide 16 Bit breit). Dabei wird die Wertemenge des einen Typs auf die des anderen abgebildet, ohne daß irgendeine Anpassung stattfindet – es kommt also ganz darauf an, wie die Werte der beiden Typen auf Maschinenebene dargestellt werden.

Man muß sich besonders den Unterschied zwischen Typ-Konversionen und Typ-Transfers bewußt machen:

```
VAR card:        CARDINAL;
    ch:          CHAR;
ch:='A';
card:=ORD(ch);
card:=65;
ch:=CHR(card);
ch:=CHAR(card);
```

ORD('A') wandelt den Typ CHAR (Zeichen) in den Typ CARDINAL (ganze Zahl ohne Vorzeichen) um, unabhängig davon, wie beide implementiert sind. Genauer gesagt, es erfolgt gegebenenfalls eine Anpassung des Bytes 41H an das Wort 0041H.

Ebenso macht CHR(65) aus dem Wort, dessen Wert sich nur in dem gültigen Unterbereich des Typs CHAR bewegen darf, wieder ein Byte. Die Formulierung CHAR (card) unterstellt währenddessen strenggenommen, daß Zeichen in 16-Bit-Worten gespeichert werden. Dabei ist unklar, was passiert, wenn die oberen 8 Bit nicht Null sind (sofern wir immer noch vom ASCII-Code ausgehen, der 8 oder eigentlich nur 7 Bit zu seiner Darstellung benötigt).

Allgemein gilt der Ratschlag, Low-Level-Konzepte (maschinennahe Konzepte) einschließlich Typ-Transfers möglichst zu meiden, wenn Programme vom Quelltext her eindeutig und portabel sein sollen. Wo sie wirklich benötigt werden, sollte man ihre Anwendung in bestimmten, als implementierungsabhängig gekennzeichneten Modulen verbergen und diese sorgfältig kommentieren, um klarzustellen, was sie eigentlich tun sollen. Zum Beispiel würde es sich in Modula anbieten, Bitmanipulationen auf bestimmten Datentypen wie CHAR mit entsprechenden Operationen in ein Modul zu packen.

Modula ist gerade für die in diesem Heft behandelten Supermikrocomputer mit objektorientierter grafischer Benutzeroberfläche wichtig. Man möchte ja seine eigenen Programme in die Fenster- und Maustechnik einbinden. Die betreffenden Softwareschnittstellen sind in der Regel extrem kompliziert, und keine Sprache als Modula-2 ist besser geeignet, um sie zu vereinfachen und ihren Gebrauch sicherer zu machen.

Wunschzettel an Modula-2

Wie bei jeder Sprache gibt es eine Reihe von möglichen Wünschen, die Modula nicht erfüllt. Obwohl wir uns der damit verbundenen Schwierigkeiten bewußt sind, würden wir es doch für wünschenswert halten, wenn auch die Low-Level-Konzepte der Sprache, zumindest für einen bestimmten Prozessortyp, sorgfältiger und einheitlicher konzipiert würden. Beispielsweise sollte, wenn der Typ »WORD« (der typischerweise für ein 16-Bit-Maschinenwort steht) um den Typ »BYTE« (8 Bit) ergänzt ist, BYTE alle Eigenschaften von WORD besitzen. Insbesondere auch die, daß ein offenes ARRAY OF BYTE als formaler Prozedurparameter ebenso wie ARRAY OF WORD jeden beliebigen Datentyp akzeptiert, um die Sprache orthogonaler zu halten. Die Low-Level-Eigenschaften von Datentypen sollten eindeutig und einheitlich sein. Wenn CHAR durch ein BYTE repräsentiert wird, sollte es sich immer so verhalten, und nicht manchmal so, als würde es durch ein Wort ausgedrückt, selbst wenn dies auf Maschinenebene aus technischen Gründen der Fall ist.

Das größte Problem ist derzeit gewiß der fehlende verbindliche und genügend allgemeingültige Standard für die wichtigsten Bibliotheken.

Auch der Sprache selbst gegenüber kann man zahlreiche Erweiterungswünsche vortragen. So zeigt eine Implementierung zum Beispiel, daß es durchaus möglich ist, das in der Handhabung einfachere generische READ und WRITE von Pascal auch in einem Modula-Compiler zu implementieren. Die Ein/-Ausgabe-Anweisungen werden dabei sinngemäß in (verdeckte) Importe und Prozeduraufrufe des Moduls »Texts« übersetzt.

Ferner vermißt man »große Mengen« (mindestens 256 Elemente, um uneingeschränkt SET OF CHAR implementieren zu können), welche der Report nicht verbietet. Es ist ein Trend zu beobachten, daß die meisten Implementierungen dies anbieten. TDI-Modula auf dem Atari erlaubt sogar Mengen mit 65000 Elementen.

Die offenen Arrays (ARRAY OF CHAR) könnten mehrdimensional sein. Auch hier gibt es bereits entsprechende Implementierungen.

Manchmal würde man sich strukturierte Funktionsresultate (ARRAYS, RECORDS) wünschen. Auch dies würde die Sprache orthogonaler gestalten.

Eine Modula-Implementierung enthält EXCEPTIONS (Ausnahmebedingungen) zur Fehlerbehandlung, wie sie von Ada bekannt sind. Das ist allerdings eine bedeutende Spracherweiterung, sie fügt sich aber recht harmonisch in das syntaktische Konzept der Sprache. Es scheint, daß Modula durch die Konzepte des Moduls und der parallelen Prozesse eigentlich eine solche Fehlerbehandlungsmöglichkeit erfordert, da alle anderen Möglichkeiten mit Fehlerzuständen und Fehlerparametern entweder sehr umständlich oder unsicher oder beides sind.

In manchen Punkten (große Mengen, opaker Export komplexer Typen) wird man sich wahrscheinlich im Laufe der Zeit darauf verständigen, die in diesen Punkten recht unverbindlich gehaltene Sprachdefinition nach oben auszudehnen. Bei anderen wie Exceptions, mehrdimensionale offene Arrays wird man sehen, ob die Praxis sich mit entsprechenden Forderungen durchsetzen wird – doch droht hier schon wieder ein neuer Wirrwarr unterschiedlich erweiterter Implementierungen. Sicher aber fordert Modula weit weniger zu willkürlichen und tiefgreifenden Spracherweiterungen und Veränderungen heraus als Pascal. Gegenwärtig ist es sogar so, daß viele Implementierungen hier und da Abstriche gegenüber Wirths Definition machen, weil die jeweiligen Implementierungen noch nicht so weit gediehen sind. (le)

Sprachmerkmal	Basic	Pascal	Modula	Ada	C
			Datentypen und zugehörige Konzepte		
Einfache Datentypen	ja	ja	ja	ja	ja (Integer, Real, Char)
Einfache Datenstrukturen	ja	ja	ja	ja	ja (Arrays ein- und mehrdimensional)
String-Standard Datentyp	ja	in bestimmten Impl.	als Array	ja	(wie UCSD/Turbo)
Stringmanipulationen	ja	in bestimmten Impl.	in entsprechr Modulen	–	nein (ev. als Bibl.)
Offene Arrays als Parameter	-	ja	ja	ja	nein
mehrdimensionale Open-Arrays	–	?	nein (nur bestimmte Implemen.)	?	nein
Abstrakte Low-Level-Typen ohne Struktur und Grenzen	–	nein	ja	?	Pointer (Array of WORD)
Komplexe Datenstrukturen (Records mit beliebigen Komponenten)	nein	ja	ja	ja	ja
Gepackte Datenstrukturen	nur String	ja, aber impl.-abhäng.	nein Array of Char je nach Implem.		nein, nur Array of Char
Benutzerdefinierte Datentypen	nein	ja	ja	ja	ja
Unterbereichstypen	nein	ja	ja	ja	ja
Abgeleitete Typen	nein	nein	nein	ja (Dollar = NEW Geld)	nein
undurchsichtiger Export von Typen	–	nein	nur Pointer oder einfache Typen	ja	nein (opake Typen)
prozedurale Typen	nein	nein	ja	ja?	nein
Moderne objektorientierte Konzepte (Smalltalk)	nein	nein (ansatzweise be Datenstrukturen)	ansatzweise	ansatzweise	nein
Export von bestimmten Attributen eines Objektes	–	über zusammengesetzte Datenstr (nur UCSD)	über zusammengesetzte Datenstr	ja	–
Zusammenfassen von Datenobjekten und Operationen zu Klassen	–	(Clascal)	bedingt	–	nein
lokale Variable	nein	ja	ja	ja	ja
Datenkapseln (lokale Bereiche mit unbeschränkter Lebensdauer der Variablen)	-	nein (nur bestimmte Dialekte, UCSD-Units)	ja	ja	ja, trifft auf Funktionen zu
			Formulierung von Algorithmen		
strukturierte Ablaufformulierung	nur FOR-NEXT und GOSUB (in manchen Dialekten ja)	ja	ja	ja	ja
benannte Algorithmen (Prozeduren)	nein	ja	ja	ja	ja
generische Standard-Ein/Ausgabe-Prozeduren	ja	ja	nein	nein	ja
selbstdefinierte Ein/Ausgabe-Prozeduren (Bibliotheken)	nein	nein	ja	ja	ja
selbstdefinierte generische Prozeduren	nein	nein	nein	ja	nein
strukturierte Funktionsresultate	nein	nein (außer bestimmte Dialekte)	nein	ja	nein
Maschinennahe Operationen	nur Peek, Poke	nein	ja	ja	ja! (Spezial-Gebiet von C)
Typ-Umwandlung	sehr beschränkt	sehr beschr. Record-Varianten. (Typ-Transfer in Turbo)	ja (kontrolliert)	ja	ja (unkontr.)
parallele Prozesse	nein	nein (außer bestimmte Dialekte, UCSD)	ja, allgemein	ja, Rendezvous	nein
			Übergeordnete Programmentwicklungs-Konzepte		
Getrennte Übersetzung	nein	nein (außer bestimmte Dialekte, UCSD)	ja	ja	ja (Module, Bibliotheken)
Versionskontrolle zwischen Modulen zur Linkzeit	–	nein	ja	?	nein
Getrennte Definition von Bibliotheksschnittstellen und Implementierung, Syntaxprüfung bei Modulen	–	nein (außer UCSD)	ja	nein	nein
Ausnahmebedingungen (Exceptions)	je nach Interpreter (On Error Gosub)	nein	nein (nur bestimmte Implementierungen)	ja	nein
Ausdrückliche Beschreibung der implementierungsbedingten Eigenschaften von Datentypen	nein	nein	nein	ja	nein
Integriertes Programmentwicklungs-System (Compiler, Editor, Debugger)	je nach Interpreter	nein (nur Turbo und UCSD)	nein, nur bestimmte Implementierungen (z.B. TDI Interface-Techn.)	im Entwurf (APSE)	nein
Unterstützung logischer Konzepte (Listen, Graphen, Prädikate)	nein	in elementarer Form (Datenstrukturen durch Pointer)			
Qualifikation	einfache maschinenunabhängige Programmiersprache	moderne, aber einfach strukturierte Sprache	moderne strukturierte Sprache, auch für Systemprogrammierung	sehr umfangreiche Systemsprache	besonders maschinennahe Sprache

»C« – Symbiose von Assembler und Hochsprache

Langsam wird es für die Heimcomputer-Freaks Zeit, von lieben Gewohnheiten Abschied zu nehmen. Denn die klassenlose Freak-Gesellschaft steht uns bevor! Warum?

Bisher zerfiel die Gruppe der Freaks in zwei Kategorien: Diejenigen, die in Basic programmieren können und auf die oftmals sehr beschränkten Fähigkeiten der mitgelieferten Basic-Interpreter ihrer Maschinen angewiesen waren. Auf der anderen Seite diejenigen, die sich in die Höhen (oder Niederungen, wie man's eben sieht) der Assemblerprogrammierung wagten, damit volle Kontrolle über ihr System hatten und den reinen Basic-Hackern allemal die Schau stahlen.

Diese Barriere zwischen Hochsprachlern und Maschinensprachlern reißt nun eine Programmiersprache nieder, die aufs Engste mit den neuen Supercomputern auf der Basis des Prozessors Motorola 68000 verknüpft ist. Die Sprache heißt »C«; die Maschinen sind: Der Atari ST und der Commodore Amiga (und in Bälde sicherlich noch weitere von anderen Hersteller).

Lange, bevor es ein ernstzunehmendes Basic für den ST gab, stellte Atari Software-Entwicklern bereits ein Entwicklungssystem zur Verfügung, das auf der Sprache C basierte. Und das ist gut so. Denn was erwartet der Profi von einer Entwicklungssprache?

Systemprogrammierer wollen möglichst nahe an die Maschine ran. Mit die ersten Programme für den ST, die in der Public Domain herumgeisterten, waren RAM-Disk-Programme (bei dem üppigen Speicherangebot des 520ST+ kein Wunder). Solche Programme setzen sehr systemnahe Operationen voraus, denn es müssen Vektoren »verbogen« werden.

Das erfordert eine Sprache, mit der jede »Bit-Fummelei« möglich ist, mit der alle Systemvariablen, alle Calls des Betriebssystems offen zugänglich sind. Assembler, denken Sie?

Nein: C! Anwendungsprogrammierer wollen die Vorzüge von GEM auf dem Atari nutzen. Nun ist die Grafikprogrammierung unter GEM kein Kinderspiel, wie andere Beiträge in diesem Heft zeigen. Zwar relativ bequem, weil dem Programmierer vom GEM-System vieles abgenommen wird (Kontrolle der Maus, Window-Verwaltung, Umgang mit grafischen Objekten), aber die Programme selbst werden anspruchsvoller. Da möchte sich nun der Programmierer nicht mehr um jedes Detail kümmern müssen. Kurzum, es verlangt nach einer Programmiersprache, die es erlaubt, durch strukturiertes Vorgehen mit den komplexen Problemen fertig zu werden, nach einer modernen, strukturierten Sprache. Pascal, denken Sie?

Nein: C! Atari hat diese Sprache mit Vorbedacht gewählt, da sie in der Lage ist, die Bedürfnisse sowohl der System- als auch der Anwendungsprogrammierer bestens zufriedenzustellen. Bloß:

Wer hat von C schon gehört? Hierzulande die Allerwenigsten; bisher war alles Basic, Assembler und wenn's hoch kam, ein bißchen Pascal. Und da mögen es, vor ihren Maschinen in Ehren ergraute Assembler-Hacker, vielleicht als Zumutung empfinden, daß man sie jetzt mit einer neuen Sprache behelligt.

Aber das sollten Sie wirklich nicht so eng sehen. Denn die neueste Entwicklung auf dem Heimcomputer-Markt, also Maschinen von der Klasse des Atari, bringt nicht nur die »Kleinen« näher an die »ausgewachsene« Computerwelt heran, (sprich: sie werden immer professioneller); sie bringt auch ihre Besitzer (jedenfalls die, die in die Programmierung eingestiegen sind) näher an den Stellenmarkt. Wenn Sie Basic und Maschinensprache des C64 voll im Griff haben, dann demonstriert das sicher, daß Sie eine Ahnung von der Materie haben. Aber nach dem Basic und Assembler des C 64 kräht auf dem Stellenmarkt für Profi-Programmierer kein Hahn mehr.

Gerade Unternehmen, die sich auf die Entwicklung von Software für Mikro-Computer spezialisiert haben, stellen zunehmend auf C um. Einmal, weil es gleichermaßen System- wie Anwendungsprogrammierung erlaubt. Zum anderen, weil Programme in C weitgehend portabel sind. Das bedeutet, daß die Übertragung eines für einen Computer »A« entwickelten C-Programms auf einen Computer »B« sich meist ziemlich problemlos gestaltet. Versuchen Sie mal, ein Programm im Basic des Sinclair ZX-Spectrum auf den Atari 800 zu übernehmen, dann wissen Sie, was mit »portabel« gemeint ist und warum das eine so schätzenswerte Eigenschaft ist!

Um zu ergründen, wie C zu diesen fabelhaften Eigenschaften kommt, ist ein kurzer Blick in die Geschichte, in die Entwicklung dieser Sprache, erforderlich.

C ist das Werk eines einzigen Mannes und nicht etwa eines hundertköpfigen Komitees (wie es zum Beispiel bei den Sprachmonstern Cobol und neuerdings ADA der Fall ist): Dennis M. Ritchie. Dieser war in den 70er Jahren maßgeblich an der Entwicklung des Betriebssystems UNIX beteiligt. Dabei fand er, daß keine der bestehenden Sprachen für seine Zwecke geeignet war. Die Entwicklung eines Multi-User- und Multi-Tasking-Betriebssystems ist kein Kinderspiel und in Maschinensprache eine schiere Quälerei; eine strukturierte Sprache wäre da eine große Hilfe. Andererseits war mit den damals vorhandenen Hochsprachen keine so maschinennahe Programmierung möglich, wie sie ein Betriebssystem erfordert. Da müssen schließlich Geräte getrieben, absolute Adressen der Maschine manipuliert und jede Menge Bit-Operationen vorgenommen werden.

Sprache für alle

Deshalb schuf Ritchie die Sprache C. Und deshalb ist beinahe das gesamte Unix (nur ein harter Assembler-Kern von zirka 5 Prozent des gesamten Systems verbleibt) und seine gesamte Systemsoftware (Editoren, Compiler, Tools) in C geschrieben; bis dahin hätten es die meisten Programmierer für unmöglich gehalten, daß so etwas in einer Hochsprache möglich ist. Man könnte jetzt meinen, C sei untrennbar mit Unix verbunden und deshalb nur für Minis und Großrechner geeignet. So denken denn auch viele, daß Heim- und Personal Computer mit ihren »Bronzezeit«-Betriebssystemen nicht C-tauglich sind. Dies ist jedoch ein Irrtum. D. Ritchie hatte nämlich die Weitsicht, C völlig unabhängig von einem speziellen Betriebssystem und den Sprachumfang von C klein zu halten: es kennt nur 28 Schlüsselwörter (den BASIC-Befehlen vergleichbar), die Sie in Bild 1 sehen können. Wegen seines kleinen Umfangs ist C leicht auf andere Maschinen zu übertragen. Und es ist standardisiert. C dürfte wohl die einzige Hochsprache sein, die in gleicher Weise auf Maschinen jeder Größenordnung verfügbar ist: vom C64 bis hin zu den Supercomputern von Cray Research. Das hebt sie deutlich vom Mitkonkur-

renten Pascal ab. Aus diesem Grunde setzte sich C auch in allen bedeutenden Softwarehäusern als Umgangssprache durch: Microsoft, Digital Research, Lotus, Ashton Tate und jetzt eben auch die Hersteller Atari und Commodore. Ein kleiner Sprachumfang bedeutet, daß die Sprache ohne allzugroßen Aufwand auf verschiedene Maschinen implementiert werden kann. Ein kleiner Sprachumfang kann jedoch auch einen Schwachpunkt darstellen. Denn der Sprachentwickler muß bei der Auswahl der Sprachkonstrukte, die er mit aufnehmen will, eine glückliche Hand haben; die Sprache muß trotz ihres beschränkten Umfangs vollständig sein, der Entwickler darf nichts Wichtiges vergessen. Wohin das führt, wenn eine »kleine« Sprache unvollständig ist, hat man ja an Basic gesehen. Dessen Sprachumfang ist zwar auch relativ klein; doch meist beginnt man schon nach kurzer praktischer Arbeit in Basic das ein oder andere Nützliche zu vermissen. Deshalb hat jeder Heimcomputer-Hersteller sein Basic etwas »aufgemöbelt«, läßt zum Beispiel lange Variablennamen zu, oder führt strukturierte Sprachelemente, wie die WHILE-Schleifen, ein – und aus ist's mit der Kompatibilität.

Was braucht der Mensch?

Ritchie hatte, was die Festlegung des Sprachumfangs betrifft, diese glückliche Hand; wohl auch deshalb, weil er kein Theoretiker, sondern in erster Linie Programmierer-Profi ist. Deshalb wußte er, was der Mensch braucht....

Bei so vielen Vorteilen muß es natürlich auch Nachteile geben, denn nichts auf der Welt ist vollkommen. Der gravierendste Nachteil von C ist seine schwere Erlernbarkeit. Das kommt nicht etwa daher, weil C besonders exotisch wäre, wie es etwa die KI-Sprachen Lisp oder Prolog sind, die vom Programmierer völliges Umdenken verlangen. C ist vielmehr ziemlich konventionell und weist starke Ähnlichkeiten zu Sprachen wie Pascal oder Algol auf. Doch es nimmt, außer einer Syntax-Überwachung durch den Compiler, keinerlei Fehlerüberprüfung vor. So steht Ihnen nichts im Wege, munter über Arraygrenzen zu schreiben. Sie können Zeiger in lebenswichtige Teile des Betriebssystems setzen und diesem den Gnadenstoß versetzen. Sie dürfen auch Zahlen, Zeichen, Adressen und vieles mehr beliebig mischen, kurz: (fast) alles ist erlaubt.

int	extern	else
char	register	for
float	typedef	do
double	static	while
struct	goto	switch
union	return	case
long	sizeof	default
short	break	entry
unsigned	continue	auto
if		

Tabelle 1. Die Schlüsselwörter von C

Nun lautet eines der Grundgesetze der Programmiererei: Der Anfänger macht jeden Fehler, der sich machen läßt. Da sich in C relativ leicht fatale Fehler begehen lassen, die zu Programmabstürzen führen, ist die Fehlerdiagnose besonders schwer. Hier hilft nur eines: Man muß sich einen besonnenen, strukturierten Programmierstil angewöhnen und so systematisch testbare Programme entwickeln. C setzt also ein gewisses Maß an geistiger und moralischer Reife beim Programmierer voraus; die Zeiten der Spontanhackerei sind erst mal vorbei!

Daß C so schwer zu erlernen ist, ist die Kehrseite der Medaille »Maschinennähe«. Nur der Grund, weil der Compiler alles zuläßt, ermöglicht auch alle maschinennahen Operationen, die für die Systemprogrammierung notwendig sind.

Der zweite Nachteil von C ist ebenfalls die Kehrseite einer Medaille. Hier geht es um die Geschwindigkeit. C-Programme sind schnell; auf jeden Fall schneller als vergleichbare Basic-Programme, oftmals ebenso schnell wie Maschinenprogramme. Dies liegt daran, weil C eine Compilersprache ist. So erweist es sich als etwas umständlich in der Handhabung – vor Gebrauch schütteln –. Sie wissen ja: Außer mit Nullen und Einsen mag Ihr Computer mit Ihnen nicht reden. Man kann dieser Binär-Sprache, dem Maschinencode, die Schärfe nehmen, indem man den Bitfolgen sprechende Namen verpaßt (Mnemonics) und diese per Programm in die Maschinensprache übersetzen läßt. Dies ist das Prinzip der Assembler.

Als nächstes kamen die Informatik-Pioniere auf den Dreh, daß man sich noch einen Schritt weiter von der Muttersprache des Prozessors wegbewegen kann: Indem man schlauere Übersetzungsprogramme baut, versteht die Maschine auch höhere Sprachen.

Simultan-Dolmetscher

Basic ist wohl das bekannteste Beispiel für eine höhere Programmiersprache, C ein anderes. Aber denken Sie stets daran: Um mit einer höheren Sprache arbeiten zu können, benötigt man stets ein Übersetzungsprogramm aus dieser höheren Sprache in die Maschinensprache. Der Bau dieses Übersetzers fordert zwei grundsätzliche Optionen. Sie lassen sich durchaus mit menschlicher Übersetzungstätigkeit vergleichen, nämlich mit der des Simultan-Dolmetschers, beziehungsweise des literarischen Übersetzers. Der Simultandolmetscher beeindruckt durch seine Fähigkeit, quasi fast gleichzeitig zum Sprecher dem Hörer die Übersetzung mitzuteilen. Dies ist analog dem Vorgehen eines Interpreters bei Basic: Der Interpreter kann ein Programm oder Programmteile unmittelbar, nachdem er sie »gehört« hat, zur Ausführung bringen. Der Vorteil dieses Verfahrens ist das prompte Feedback: Sowohl beim menschlichen Simultandolmetscher als auch beim Programmiersprachen-Interpreter erkennt man sofort an den Reaktionen des »Gesprächspartners«, ob man das Richtige gesagt hat. Aber zurück zum Simultandolmetscher; die Qualität der Übersetzung leidet natürlich unter dem Zwang zur unmittelbaren Übertragung. Wenn jemand besonders treffend und prägnant formuliert, kommt der Simultandolmetscher ins Schwimmen. Doch stellen Sie sich mal vor, man würde von einem Dolmetscher verlangen, ein Shakespeare-Sonett simultan zu übersetzen. Sehen Sie? Dafür braucht man die literarischen Übersetzer. Und die brauchen Zeit, weil sie im stillen Kämmerlein Wörterbücher wälzen, an Formulierungen herumbasteln und die Ideen reifen lassen müssen. Im EDV-Sektor entsprechen die Compiler den literarischen Übersetzern. Bei ihnen verstreicht zwar zwischen der Programmeingabe und der Programmausführung mehr Zeit als beim Interpreter; aber die Übersetzungsergebnisse sind meist besser, laufen schneller oder brauchen weniger Platz. Wenn's um Qualität geht, muß der Compiler ran.

Der Compiler macht's

Das benötigt aber einiges an Umständen. Das Arbeiten mit einem C-Compiler spielt sich so ab: Mit einem Editor erstellen Sie in einer Datei das Programm (dieses in der Datei stehende Programm nennt man auch »Quellprogramm«). Meist ist der Name der Datei nach Durchlaufen der gesamten Übersetzungsprozedur mit dem Namen des ausführbaren Programms (des »Objektprogramms«) identisch. Danach kommt der Compiler dran: Er übersetzt das Quellprogramm allerdings bei den meisten C-Compilern zuerst in Assembler-Mnemonics. Das hat den Vorteil, daß der Programmierer laufzeitkritische Teile seines Programmes »handoptimieren« kann, indem er sich den vom Compiler erzeugten Code betrachtet und Ungeschicklichkeiten in der Über-

setzung ausbügelt. Es gibt jedoch heutzutage bereits optimierende Compiler von solcher Qualität, daß die Handoptimierung so gut wie überflüssig ist – allerdings gibt's die noch nicht für den Atari! Aber die meisten Anwendungen erfordern ohnehin keine Handoptimierung.

Die Assembler-Mnemonics, die der Compiler erzeugt hat, müssen von einem Assembler weiterbehandelt werden. Wer jetzt meint, das Ergebnis des Assemblers sei sein fertiges Programm, der irrt. Das liegt in der Modularität von C begründet (siehe unten): C erlaubt es, Programmteile separat zu compilieren und für den späteren Gebrauch in Bibiliotheken zu plazieren. Was bei Modula-II jetzt so bejubelt wird, ist in C also ein alter Hut! Wenn Sie Tabelle 1 betrachten, die den gesamten Sprachumfang von C wiedergibt, dann wird Ihnen auffallen, daß keinerlei Anweisungen für Ein-/Ausgabe, für trigonometrische Funktionen oder Stringverarbeitung in C enthalten sind. Das bedeutet aber nicht, daß so etwas nicht geht. Die mit jedem C mitgelieferte Bibliothek enthält alle dazu nötigen Funktionen. Dieses Verfahren hat unter anderem den Vorteil, daß man die Bibliothek um eigene Ein-/Ausgabefunktionen, Stringfunktionen etc. erweitern kann. Eine Liste der Operatoren zeigt Tabelle 2.

Die meisten Programme benutzen also Module aus der Bibliothek, die erst noch mit dem Programm verknüpft werden müssen. Dazu ist der Linker da; er nimmt das Assembler-Ergebnis und sieht nach, welche Funktionen dem Programm noch fehlen. Dann durchsucht er die ihm angegebenen Bibliotheken nach den vermißten Modulen, klebt alles zusammen und erzeugt – endlich, endlich! – ein lauffähiges Programm. Fehler können in diesem Prozeß an zwei Stellen auftreten: Einmal können sich in Ihr Programm Syntaxfehler eingeschmuggelt haben, die dann der Compiler erkennt. Oder aber Sie verwenden ein Bibliotheks-Modul im Programm, das es gar nicht gibt (sowas passiert zum Beispiel dann, wenn man sich verschreibt und statt »printf« fälschlich »pintf« tippt). Diesen Fehler erkennt der Compiler nicht; erst der Linker wird ihn anmahnen. Und wenn Sie dann mit der ganzen Prozedur fertig sind, dann finden sich natürlich noch die allseits unbeliebten Logikfehler im Programm – und der ganze Zyklus geht von vorne los.

Das ist lästig; wer aber wirklich ernsthaft mit C arbeitet, der kann diese Prozedur auch als ausgesprochen läuternd erfahren. Man überlegt sich nämlich dann genau, was man da zusammenprogrammiert; denn für jeden Fehler hat man (bei einem durchschnittlich schnellen Compiler und wenn man mit Disketten arbeitet) mit einer durchschnittlichen Strafzeit von 5 Minuten zu rechnen! Dies führt dazu, daß man seine Programme besser durchdenkt. Um die Sache noch etwas komplizierter zu machen, ist beim C das Entwicklungssystem der Compiler auch noch in drei Schritte aufgeteilt, nämlich in Makro-Präprozessor, Parser und Codegenerator. C kann, wie ein Makro-Assembler, C-Makros vereinbaren. Das können einfach nur Konstanten sein, aber auch kompliziertere Ausdrücke mit Funktionscharakter, die mit Parametern versehen sind, kommen vor. In den Beispielen zu diesem Artikel finden Sie mehrere Makro-Definitionen, die Sie daran erkennen, daß sie mit » # define« eingeleitet sind. Der Präprozessor ersetzt erstmals die Makros durch den sie definierenden Ersetzungstext, ehe der Compiler das Ganze zu sehen bekommt. Die nach dem Makro-Präprozessor kommenden Schritte »Syntaxanalyse« (oder »Parser«) und Codegenerierung sind bei jedem Compiler (also auch bei Pascal, Cobol, Fortran..) notwendig, und es ist lediglich eine verfahrenstechnische Frage, sie in hintereinander auszuführende Schritte aufzuspalten. Um also im Atari-C ein Programm namens »test« zu erstellen, sind folgende Schritte nötig:

1. Mit dem Editor die Quelldatei »test.c« erstellen.
2. Makro-Präprozessor rufen: CP68 test.c test.i
3. Parser rufen: C068 test. I test.1 test.2 test.3
4. Codegenerator rufen: C168 test.1 test.2 test.s
5. Assembler rufen: AS68 test.s
6. Linker rufen (»gems«, »apstart«, »osbind«, »gemlib« und »libf« sind Bibliotheken): LINK 68 test.68k=gems,apstart,test,osbind, gemlib,libf

Wie Sie sehen können, erzeugen die einzelnen Schritte des Compilers jede Menge Zwischendateien; da wird die Diskette ganz schön voll! Neben dem C des Entwicklungssystems wird zur Zeit noch ein anderes C für den ST angeboten, das von der Firma GST stammt. Während das Atari-C kommandoorientiert ist (es wurde ursprünglich für den Einsatz unter CP/M-68K entwickelt), nutzt das GST-C die Vorteile von GEM. Gerade für Anfänger ist dieses C besonders einfach zu bedienen, da es durch ein Steuerprogramm – die sogenannte Shell – selbsttätig nacheinander alle nötigen Schritte (Editor, Compiler, Assembler, Linker) aufruft und deren Ergebnis protokolliert. Angenehm fällt bei diesem C auch auf, daß ein Editor

Einstellige Operatoren – Erklärung	
*	liefert das Objekt, auf das eine Adresse zeigt
&	liefert die Adresse eines Objekts
-	liefert den Kehrwert einer Zahl
!	liefert das logische Komplement eines Wahrheitswerts
~	liefert das Einer-Komplement (bitweise Negation)
++	Prä- und Postinkrement
--	Prä- und Postinkrement
sizeof	liefert die Größe eines Objekts in Bytes
Zweistellige Operatoren	
*, +, -, /	die übliche Arithmetik
%	Divisionsrest (Modulo-Division)
»,«	Shift rechts beziehungsweise links
<, >,	
<=, >=	Vergleichsoperatoren
==	Test auf Gleichheit
!=	Test auf Ungleichheit
&	bitweises UND
^	bitweises XOR
|	bitweises ODER
&&	logisches UND
||	logisches ODER
Dreistellige Operatoren	
?:	IF-THEN-ELSE als Operator
Zuweisungs-Operatoren	
=	»Normale« Zuweisung
+=, -=,	Zuweisung mit gleichzeitiger Addition, Subtraktion
*=, /=, %=	Zuweisung mit Multiplikation, Division und Modulo-Division
»=, «=	Zuweisung mit gleichzeitigem Shift
&=, ^=, |=	Zuweisung mit gleichzeitigen bitweisen Operationen

Tabelle 2. Die Operatoren von C

(der ebenfalls Windowing kann und mit der Maus spricht) zum Lieferumfang gehört.

Geht man bei C ins Detail, so sind drei Gesichtspunkte zu beachten: die Modernität (die es als strukturierte Sprache auszeichnet), die Maschinennähe, die es zur Systemprogrammierung eignet, sowie die Möglichkeit, sich in C unübertroffen konzis – oder, wie Kritiker sagen: verklausuliert – auszudrücken. Als »Sympathisant« zieht man es natürlich in diesem Zusammenhang vor, von der Möglichkeit zu sprechen, in C »hochintegrierte Ausdrücke« zu schreiben.

»C« zerlegt

Die Modernität von C zeigt sich in drei wichtigen Aspekten: der Modularität von C, den leistungsfähigen Kontrollstrukturen und der Möglichkeit zur Bildung beliebig komplexer Datenstrukturen. Stichwort Modularität: C verfolgt das Funktionskonzept mit Blockstruktur und lokalen Variablen; Funktionen können in separat compilierten Modulen abgelegt werden ... halt, halt! Alles schön der Reihe nach. C-Programme sind Funktionen, die gerufen werden und eventuell einen Wert produzieren. Ein Beispiel dafür sehen Sie in Bild 1. Dies ist ein kleines C-Programm, das »Hallo, lieber Happy-Computer-Leser!« druckt.

Interessant ist nicht das Ergebnis (das ist sogar ziemlich fade), sondern die Art, in der es erreicht wird. Im Programm werden drei Funktionen definiert, die die Namen »main«, »hallo« und »leser« tragen. Mit der Funktion »main« hat es eine besondere Bewandtnis: sie ist das eigentliche Programm, das Hauptprogramm (daher auch der Name!). Das, was »main« macht, macht auch Ihr Programm; und jetzt müssen Sie nur noch wissen, was »main« in Bild 1 eigentlich tut.

Das steht im Anweisungsblock der Funktion, dem in geschweiften Klammern stehenden Teil, der sich an den Funktionsnamen anschließt.

Im Anweisungsblock können eine oder mehrere Anweisungen stehen, die mit Strichpunkt abgeschlossen werden müssen und normalerweise in der Reihenfolge ausgeführt werden, in der sie dastehen. Im Beispiel sind es zwei Funktionsaufrufe, kenntlich an den runden Klammern. Die Funktionen, die gerufen werden, sind selbstdefiniert und ihre Definition schließt sich an die von »main« an.

Das muß aber nicht so sein (nämlich daß die Definition einer Funktion in der gleichen Datei wie das Hauptprogramm zu finden ist). Es ist – wie bereits erwähnt – ebensogut möglich, daß Funktionen, die ein Programm benutzt, in der Bibliothek enthalten sind.

Eine Funktion von allgemeinem Wert kann getrennt compiliert und als Modul in der Bibliothek abgelegt werden. Bei den Funktionen »hallo« und »leser« aus dem Beispiel lohnt sich das natürlich nicht!

Aber in »hallo« und »leser« ist eine solche Funktion, die aus der Bibliothek stammt, enthalten: »printf« nämlich, das eine Meldung (einen String) auf dem Bildschirm ausgibt (und das sehr vielseitig ist, wie Sie noch sehen werden). Das modulare Konzept nimmt der Arbeit mit dem Compiler ein wenig die Schärfe, weil man Module separat entwickeln, austesten, übersetzen und in die Bibliothek stecken kann. Sie müssen also nicht bei jedem Programm extra neu übersetzt werden; das spart Zeit.

Funktionen können aber auch Argumente haben und einen Wert zurückgeben. Das Programm in Bild 2 ist eine etwas umständliche Art, um die Summe, die Differenz und das Produkt zweier Zahlen zu berechnen. Sie zeigt Mehreres: Daß »printf« nicht nur Konstanten, sondern auch berechnete Werte ausgeben kann; daß Funktionen Argumente haben und daß Variable in C lokal sind.

Um »printf« dazu zu bewegen, berechnete Werte (oder auch Variable) auszugeben, muß man in den String Steuerzeichen (mit einem Prozentzeichen eingeleitet) einstreuen, die angeben, was für Werte man wo erhalten will. Hinter dem String muß dann für jedes Steuerzeichen ein auszugebender Wert an »printf« übergeben werden; im Beispiel ist es immer nur einer. Die Zeichenfolge »\n« zu Beginn des »printf«-Kontrollstrings sorgt übrigens für einen Zeilenvorschub (Abkürzung für englisch »newline«; daneben gibt es unter anderem noch »\t« für Tabulator oder »\b« für Backspace).

Funktionen können in C Argumente haben; dann muß man in der Definition der Funktion angeben, wieviele Argumente sie hat und von welchem Typ sie sind. Diese Parameter-Deklaration schließt sich an den Funktionsnamen an.

```
main()
{   hallo();
    leser();
}

hallo()
{   printf("Hallo, ");
}

leser()
{   printf("lieber ");
    printf("Happy-Computer-Leser!");
}
```

Bild 1. C-Funktionen (ganz ganz einfache...

Die ersten beiden Funktionen arbeiten auch noch mit einer Variablen, in der sie das Ergebnis speichern. Auch Variable müssen deklariert werden: Man sagt damit dem Compiler, wieviel Speicherplatz er für die Variable besorgen soll; das ist bei den einzelnen Datentypen von C unterschiedlich. Auf dem Atari benötigen die Zeichen (in C »char« genannt) ein Byte, die ganzen Zahlen mit einfacher Genauigkeit (C: »int«) zwei Byte, mit doppelter Genauigkeit (»long«) 4 Byte, also ebenso viel wie einfach genaue Gleitpunktzahlen (»float«). Außerdem gibt es noch doppelt genaue Gleitpunktzahlen (»double«) mit 8 Byte.

Im Bild 3 haben Parameter und Variable in den beiden Funktionen »sum« und «diff« die gleichen Namen. Aber das führt zu keinem Konflikt. Die Variable »ergeb« in »sum« ist eine ganz andere als die Variable »ergeb« in »diff«. Dies zeigt, was mit »Lokale Variable« gemeint: Eine Variable kann nur innerhalb der Funktion verändert werden, in der sie definiert ist. Außerdem zeigt es, daß Variablennamen – ebenso wie die Namen für selbstdefinierte Funktionen – in C lang und damit sprechend sein dürfen (»ergeb«). Auch die dem Basic-Programmierer gewohnten globalen Variablen (die überall im Programm bekannt sind) gibt es in C; allerdings ist da die Ausnahme, was in Basic die Regel ist!

Funktionen können Werte zurückgeben; was sie zurückgeben, bestimmt »return«, die Anweisung, mit der die Funktion auch verlassen und die Kontrolle wieder an die rufende Funktion abgegeben wird (im Beispiel ist dies stets das Hauptprogramm »main«).

Die Funktion »prod« demonstriert, daß Parameter beliebige Namen haben dürfen, und daß man das alles auch noch viel kürzer erreichen kann. Die letzte Anweisung von »main« macht deutlich, wie unsäglich umständlich das Vorgehen in diesem Beispiel ist; aber was tut man nicht alles, um eine neue Sprache zu lernen!

Die Anweisungen im Anweisungsblock einer C-Funktion können Wertzuweisungen an eine Variable sein oder Aufrufe von Funktionen; für beides bietet Bild 2 Beispiele. Dann gibt es aber auch noch Anweisungen, die die Steuerung anderer Anweisungen beziehungsweise Anweisungsfolgen übernehmen: die sogenannten »Kontrollstrukturen«. Das klingt mysteriös, Sie kennen das aber schon: als Schleifen und bedingte Verzweigungen. Kontrollstrukturen heißen sie, weil sie Abweichungen von der normalen Reihenfolge der Auswertung kontrollieren; und die (die normale) ist: erst die erste Anweisung, dann die zweite Anweisung, dann

```
main()
{    int i, j,

     i = 13,
     j = 44,

     printf("\ni = %d, j · %d", i, j),
     printf("Summe· %d   sum(i,j)),
     printf( Differenz  %d , diff(i,j)),
     printf("\nProdukt: %d", prod(i, j));
     printf("\nQuotient: %d", i / j);
}

sum (x, y)
     int x, y,

{    int ergeb:

     ergeb = x + y;
     return ergeb,
}

diff (x, y)
     int x, y,

{    int ergeb,

     ergeb = x - y;
     return ergeb,
}

prod (a, b)
      int a, b;

{    return a * b;
}
```

Bild 2. Elementare Arithmetik in C

... haben Sie's? Die bedingten Verzweigungen gehen in C mit »if« beziehungsweise »if...else« und unterscheiden sich nicht besonders von dem, was man aus anderen Sprachen kennt; ein Beispiel dafür finden Sie in Bild 3.

Aber bei den Möglichkeiten zur Schleifenbildung hat C etwas Besonderes zu bieten: eine äußerst komfortable und äußerst strukturierte »for«-Schleife. Betrachten Sie dazu Bild 3, das – als auch nicht gerade besonders umwerfendes Beispiel – eine Quadratzahlen-Tabelle erstellt. Sie sehen, daß in C die Kontrollbedingung, die die Auswahl des »if« steuert, in runden Klammern stehen muß. Soll das »if« mehrere Anweisungen kontrollieren, dann werden diese mit geschweiften Klammern zu einem Anweisungsblock zusammengefaßt (weswegen C auch »blockstrukturiert« heißt). Daneben gibt es auch die aus Pascal bekannte Mehrfach-Auswahl mittels »switch« und »case«.

Aber viel interessanter ist die »for«-Schleife. Auch sie ist mit einer runden Klammer versehen, deren Inhalt es in sich hat. Bild 3 zeigt Ihnen, welches die Bestandteile einer »for«-Schleife sind. Der Witz dabei: In der runden Klammer wird angegeben, welchen Anfangswert die Schleifenvariable (hier: »i«) haben soll (im Initialisierungsteil), wie lange die Schleife durchzuführen ist (in der Kontrollbedingung), und was nach jedem einzelnen Schleifendurchlauf mit der Schleifenvariable zu geschehen hat (im Reinitialisierungsteil). Hier wird die Laufvariable »i« um Eins hochgezählt (wie das geht, können Sie weiter unten lesen). In diesem, in runden Klammern stehenden Schleifenkopf, hat man alles beisammen, was es faktisch an Informationen über die Schleife zu wissen gilt. Das steigert die Übersichtlichkeit. Übrigens ist man nicht nur auf eine einzelne Schleifenvariable beschränkt: In allen drei Teilen des Schleifenkopfs kann man mehrere, durch Komma getrennte, Initialisierungen, Kontrollbedingungen und Reinitialisierungen angeben. Neben der »for«-Schleife, die in der Terminologie der Informatik Zählschleife heißt, gibt es in C auch die, ausschließlich von logischen Bedingungen kontrollierten, »while«- und »do«-Schleifen. Erstere ist eine abweisende, letztere eine nichtabweisende Schleife. Aber Kontrollstrukturen alleine machen noch keine strukturierte Sprache. Bei einigen neueren Heimcomputern (Schneider, Commodore 128) wurde das Basic so aufgemotzt, daß es auch strukturierte Schleifen (meist WHILE) kennt; aber deswegen ist Basic noch lange keine strukturierte Programmiersprache. Dazu braucht es auch noch Datenstrukturen.

C bietet dem Programmierer die Möglichkeit, aus den einfachen Datentypen der Sprache komplexere Gebilde zusammenzubauen. Aus Basic kennt man ja die Arrays (Felder), die Aneinanderreihungen von Speicherzellen gleichen Typs sind. So etwas existiert natürlich auch in C (am Rande: Die in Basic so beliebten Strings sind in C Zeichen-Arrays, was auch den Vorteil hat, daß sie beliebig lange sein können!)

Der Weisheit letzter Schluß

Arrays gibt es in C auch in mehrdimensionaler Ausführung. Aber der Weisheit letzter Schluß sind die Arrays nicht. Denn es ist nicht möglich, in ihnen verschiedene Daten-Typen zu mischen; also zum Beispiel Zahlen, Zeichen und Strings als Einheit zusammenzugruppieren. Das übernehmen in C die Strukturen, oder, wie der Fachmann lässig sagt: die »Structures«. Die Structures sind besonders interessant, weil sie auch rekursiv sein dürfen, und somit (zusammen mit der dynamischen Speicherverwaltung) modernste Programmiertechniken erlauben. Und weil der Atari ein moderner Computer ist, ist auch seine Systemsoftware modern und arbeitet mit rekursiven Strukturen. Eine dieser Strukturen des Betriebssystems wertet den sogenannten »Buffer Control Block« (BCB) aus. Leider verfügen nicht alle C-Compiler, die zur Zeit für den Atari ST angeboten werden, über die wichtige Fähigkeit, Strukturen aufzurufen. Macht der Atari Disketten-Ein-/Ausgaben, dann werden die zu übertragenden Daten im Speicher gepuffert, um die Anzahl der Disketten-Zugriffe gering zu halten. Die BCBs sagen dem System unter anderem, wo es die zugehörigen Puffer finden kann und welche Daten darin gespeichert sind. In Bild 4 finden Sie auch eine Grafik der Struktur, die die BCBs aufweisen. An den Systemadressen 4B2 und 4B6 finden sich zwei Zeiger auf solche BCB-Strukturen. Jeder BCB weist unter anderem auf seinen Nachfolger und auf einen Puffer, der die Daten enthält. Mit den Zeigern (die der C-Programmierer auch »Pointer« nennt) lassen sich also Strukturen miteinander verketten. Die Verkettung hat den Vorteil, daß das System sich nach Bedarf weitere Puffer (mit zugehörigen BCBs) irgendwo im Speicher besorgen kann, wo es gerade Platz findet. TOS ist durch diese Flexibilität sehr effektiv in der Speicherausnutzung. Das Ansprechen dieser Strukturen unterstützt der C-Compiler von DRI sehr mit seiner unmittelbaren Maschinennähe, die man sonst nur in der wesentlich unkomfortableren Assemblersprache findet. Sie geht sogar so weit, daß man bei einem guten C-Compiler oft benötigte Variablen direkt in ein

```
main()

{    int i, anfang, ende;

     printf("\nBitte Anfangs- und Endwert eingeben: ");
     scanf("%d%d",&anfang, &ende);

     if (ende >= anfang)
     {    printf("\nSie wollen mich wohl verschaukeln...");
          return;
     }

     for (i = anfang; i<= ende; ++i)
          printf("\nDas Quadrat von %d ist &d.",i, i*i);

}


for (<initialisierung>; <kontrollbedingung>; <reinitialisierung>)
        <anweisung(en)>
```

Bild 3. Die berühmte »For«-Schleife von C

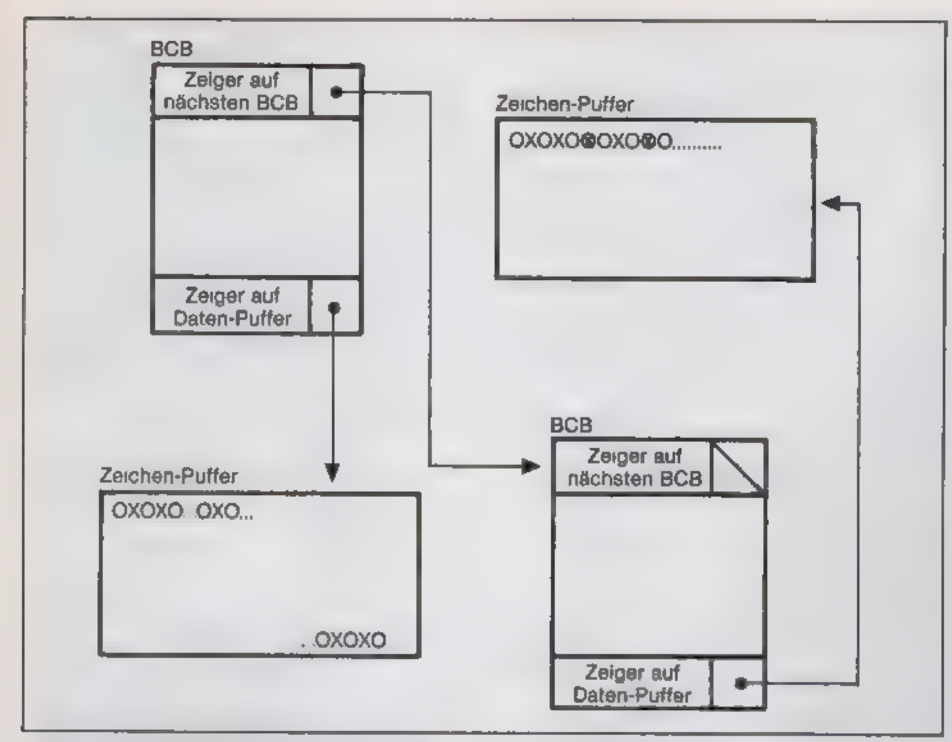


Bild 4. Die Struktur der BCB-Liste (Beispiel für eine C-Structure)

Register des Mikroprozessors ablegen kann. Allerdings kann man nicht bestimmen, in welches Register diese Variablen abgelegt werden. So tief kann man nach wie vor nur mit Assembler einsteigen. Der Zugriff auf eine solche Variable ist der schnellstmögliche überhaupt.

Für Bit-Beißer bestens geeignet

Ebenso ist es möglich, mit C Werte an absolute Speicherstellen zu schreiben. Aber für maschinennahe Programmierung reicht das nicht aus; da fehlen noch jede Menge Bitoperationen. Was das betrifft, ist C bestens sortiert. Es bietet eine geradezu überwältigende Fülle von Operatoren, wie ein Blick auf Tabelle 2 überzeugt. Darunter sind alle benötigten Bit- und Schiebeoperatoren, die man zum Setzen und Ausmaskieren einzelner Bits braucht.

In der Übersicht aller Operatoren fällt auf, daß es nicht nur einen, sondern gleich elf (!) Zuweisungs-Operatoren gibt. C bietet nämlich die Möglichkeit, mit den Zuweisungen auch gleich eine Operation zu kombinieren. Dies erscheint zuerst befremdlich, erweist sich aber mit der Zeit als ausgesprochen nützlich. Wie oft haben Sie in Basic schon sowas wie »i = i + 3« geschrieben? In C macht man das so: »i += 3;« Das schreibt sich nicht nur schneller hin, sondern ist vom Compiler auch effizienter in Maschinencode zu übersetzen.

Weil wir schon von Effizienz sprechen: C erlaubt dem Programmierer, per Deklaration gewisse Variable nicht im Arbeitsspeicher, sondern in Registern der Maschine ablegen zu lassen; dies bietet sich natürlich mit Vorliebe bei besonders hart arbeitenden Variablen an. Besonders ist dies natürlich zum Beispiel in allen Schleifen der Fall, da gerade hier immer wieder in kurzen Abständen dieselben Variablen benötigt werden. Diese Methode spart somit einiges an Zeit

Ein Spezialfall der Zuweisung hat in C wegen seiner Bedeutung eine Sonderbehandlung erfahren: Es betrifft die Erhöhung beziehungsweise Verminderung einer Variablen um Eins. Man nennt dies auch Inkrementieren (beim Erhöhen; beim Vermindern entsprechend Dekrementieren). Der Grund für diese Sonderbehandlung: Bei den meisten Prozessoren löst ein Maschinenbefehl das Inkrementieren aus. Deshalb schreibt man in C auch statt »A = A + 1« einfach »++a« oder »a++«.

Daß es zwei Schreibweisen gibt (die unterschiedliche Wirkung haben), liegt an einem weiteren wichtigen Merkmal von C: Alle Operatoren und Zuweisungen haben einen Wert, der weiterverwendet werden kann. So besitzt auch »++« einen Wert, der jedoch davon abhängt, wo es steht. Dies wird im Vergleich C/Basic wohl am schnellsten klar:

```
b = ++c;
```

entspricht

```
C = C + 1 : B = C
```

während man für

```
B = C++
```

in Basic schreiben muß

```
B = C : C = C + 1
```

Die Adreßmanipulation hängt mit dem Pointer-Konzept von C zusammen; die beiden Operatoren »*« (einstellig; bitte keinesfalls mit der Multiplikation verwechseln!) und »&« spielen hier die Hauptrolle.

Wortwitz

Erst einmal zum Stern; dieser steht nicht nur für die Multiplikation, sondern als einstelliger Operator auch noch für das, was der Informatiker »Dereferenzierung« zu nennen beliebt. Den Wortwitz, daß ein Zeichen für Zweierlei steht, leistet sich übrigens Basic auch: »=« steht für Zuweisung und Gleichheits-Test.

Aber »Dereferenzierung«? Das heißt nichts anderes, als daß man das Objekt ansprechen will, auf das eine Adresse (die Referenz) verweist. Nehmen wir an, Sie wollen in C an die Adresse 50 den Wert 13 schreiben, also

```
POKE(50,13)
```

in Basic. Ganz anders in C; da schreiben Sie

```
i = 50;
*i = 13;
```

Mit der ersten Zeile weisen Sie der Variablen »i« den Wert 50 zu. Mit der zweiten Zeile wird jedoch dieser Wert nicht überschrieben, wie man meinen könnte. Vielmehr erhält das Objekt mit der Adresse, die in »i« gespeichert ist, den Wert 13 zugewiesen. In diesem Fall bedeutet das, daß eine Speicherzelle direkt manipuliert wird.

Einfach aber wirkungsvoll

Man kann in C aber nicht nur mit Variablen, sondern auch mit anderen Dingen so umspringen. Dafür gibt's den »&«-Operator, der – vor irgend ein Objekt geschrieben – die Adresse dieses Objekts liefert. Hier ist vorsätzlich das schwammige Wort »Objekt« gebraucht, denn mit »&« kann man sich nicht nur die Adresse einer Variablen besorgen (und so zum Beispiel Zeiger aufs krauseste umbiegen), sondern auch die von Funktionen oder Unterprogrammen! Ein kleines Beispiel macht's verständlich:

```
i = 5
ip = &i;
*ip = 0;
```

Die erste Zeile ist klar. In der zweiten Zeile wird der Variablen »ip« die Adresse von »i« zugewiesen; in Zeile 3 wird diese Adresse zwecks Zuweisung dereferenziert: Als Ergebnis hat die Variable »i« jetzt den Wert 0!

Die vielfältigen Operatoren von C und die Tatsache, daß allen Operatoren ein Wert zugeordnet ist, ermöglicht es, sich ganz besonders knapp auszudrücken. Gegner von C werden hier sagen: »un-

verständlichen Code zu schreiben«. In C sieht man häufig Schleifen, in denen gleichzeitig in einer Anweisung auf das Endekriterium getestet, ein Wert besorgt, zugewiesen, die Variable manipuliert und ein Pointer erhöht wird. Sowas geht nach dem Muster:

```
while (a&= *cp++)
```

Angenommen, »cp« zeigt auf einen String. Dann wird damit das Zeichen im String, auf das »cp« gerade zeigt, besorgt (durch den »*«), der Pointer »cp« mit »++« inkrementiert und an die Variable »a« das zugewiesen, was sich durch logisches ANDen mit dem bereits vorhandenen Wert von »a« ergibt. Sollte dabei der Wert 0 herauskommen, wird außerdem die Schleife abgebrochen: All dies steckt in jener unscheinbaren Zeile!

Wenn Sie eine zeitlang C betrieben haben, dann geht Ihnen solcher Code glatt von der Hand. Aber wer dies zum ersten Mal sieht, der hält C wohl für eine Geißel Gottes, erschaffen, um unschuldige Programmierer zu strafen! Die Kritiker mögen jedoch bedenken: Oft ist diese knappe Schreibweise der Schlüssel für besonders effizienten Code. Sie enthält einiges an Hinweisen, die der Compiler für Optimierung nutzen kann.

»C« und TOS

Wie das letzte Beispiel gezeigt hat, ist dem C-Programmierer der direkte Zugriff auf die Memory-Mapped Register des Atari möglich. Das ermöglicht eine sehr systemnahe Programmierung. Doch in der überwiegenden Vielzahl der Fälle wird man sich nicht so tief in das System hinunterbegeben müssen, denn die Schnittstelle zwischen C und dem Betriebssystem ist der Sprache wie auf den Leib geschrieben. Das hauseigene Betriebssystem des Atari ist TOS; GEM ist nur eine diesem TOS aufgesetzte, grafische Benutzeroberfläche – aber was für eine! Darüber später mehr. TOS gliedert sich – wie seine bekannten Vorgänger CP/M und MS-DOS – in zwei große logische Einheiten: Einen Teil, der sich um die Logik der Kommunikation mit den Geräten kümmert (dieser weiß, was Dateien, Directories, Unterdirectories, Pfadnamen und so weiter sind), genannt GEMDOS. Zum anderen einen hardwareabhängigen Teil, der mit den Geräten sprechen kann und die Wünsche des GEMDOS in Kommandos für den Bildschirm, die Tastatur und nicht zuletzt die Disketten umsetzt. Dieser Teil heißt BIOS (ebenso wie bei CP/M und MS-DOS; die nennen »ihr« GEMDOS jedoch BDOS). Das BIOS ist wiederum getrennt in einen Teil, der das Handling der mehr konventionellen Geräte (Bildschirm, Tastatur, Drucker, Disketten) besorgt und einen Teil, der die »Extras« des Atari, die ihn so attraktiv machen, verwalten kann: Maus, MIDI-Port, hochauflösende Farb-Grafik, Sound-Generator etc. Der mehr konventionelle Geselle wird BIOS genannt, der andere XBIOS (für »Extended BIOS«).

»C« und GEM

GEMDOS, BIOS und XBIOS sind allesamt aus C heraus ansprechbar; im C des Entwicklungssystems gibt es dazu drei Bibliotheksfunktionen; die heißen – wie könnte es auch anders sein – »bios«, »xbios« und »gemdos«(!), die der Programmierer wie jede ganz normale C-Funktion einsetzen kann. Damit ist es möglich, die totale Kontrolle über das System aus C heraus zu übernehmen und es gibt auch für Systemprogramme keinen Grund mehr, zu Assembler zu greifen!

Die Funktionen des Betriebssystems wurden von den Entwicklern eigens an die in C üblichen Konventionen angepaßt. Dazu nur ein kleines Beispiel: In C werden Strings durch ein Byte mit einer binären Null abgeschlossen. Die GEMDOS-Funktion zur Ausgabe von Strings hält sich an diese Konventionen. Dies ist anders als zum Beispiel bei CP/M, dessen BDOS das Stringende an einem Dollarzeichen erkannte, was den C-Programmierer zu einigen Unbequemlichkeiten zwang. Die Programmierung mit GEMDOS demonstriert in Bild 5 ein Programm, das das Directory einer Diskette ausgibt und dabei auch Wildcards (Namensmuster mit den Jokerzeichen »?« und »*«) erlaubt. Es benutzt dazu eine selbstdefinierte Funktion »step«, die das Inhaltsverzeichnis nach passenden Namens-Einträgen durchsucht und dabei mit drei »gemdos«-Aufrufen arbeitet: »Fsetdta«, mit dem die Adresse eines Speicherbereichs an das GEMDOS übergeben wird, in den die beiden anderen Funktionen ihre Resultate schreiben. »Fsfirst«, das zur erstmaligen Suche im Inhaltsverzeichnis benutzt werden muß und »Fsnext«, mit dem man weiteres Suchen mit demselben Muster wie beim »Fsfirst«-Aufruf anstoßen kann. Dieses Vorgehen ist etwas umständ-

```
                                                  /****************************/
extern long gemdos();                             /* "gemdos" in Bibliothek   */
                                                  /*  deklarieren             */
#define TRUE (1)                                  /* Standard-Definitionen.   */
#define FALSE (0)                                 /*                          */
#define Fsetdta(ptr)  gemdos(0x1a, ptr)           /* Makros fuer Aufruf des   */
#define Fsfirst(pat)  gemdos(0x4e, pat, 0xff)     /*  Betriebssystems.        */
#define Fsnext()      gemdos(0x4f)                /*                          */
                                                  /*                          */
struct DTA                                        /* Struktur, in der Fsfirst*/
{  char schrott[30];                              /*  und Fsnext gefundene    */
   char name[14],                                 /*  Eintraege uebergeben.   */
} dta,                                            /*                          */

                                                  /****************************/
main(argc,argv) int argc; char **argv;            /* Inhaltsverzeichnis aus- */
                                                  /*  geben.                  */
{  char buff[13];                                 /****************************/
                                                  /*                          */
   ++argv;                                        /* Programm erwartet Na-    */
                                                  /*  mensmuster als Argu-    */
   while (step(*argv,buff))                       /*  ment;  step" liefert    */
      puts(buff);                                 /*  bei jedem Aufruf pas-   */
                                                  /*  senden Eintrag oder 0,  */
}                                                 /*  falls keiner gefunden.  */
                                                  /****************************/

                                                  /****************************/
step(muster,buff)                                 /* Schrittweises Durchsu-   */
    char *muster, *buff;                          /*  chen des Directory      */
                                                  /*  mit Namensmuster.       */
                                                  /*                          */
{  static int first = TRUE;                       /* Schalter: wurde "step"   */
                                                  /*  bereits einmal gerufen?*/
   switch (first)                                 /****************************/
   {  case TRUE:  Fsetdta(&dta);                  /* Erster Aufruf: Adresse   */
                  if (!Fsfirst(muster))           /*  der DTA fuer Informa-   */
                  {  strcpy(buff,dta name);       /*  tionsuebergabe setzen.  */
                     first = FALSE,               /*  Ersten gefundenen Na-   */
                     return TRUE,                 /*  men uebergeben.         */
                  }                               /*                          */
                  else return FALSE;              /*  Nix gefunden; sorry!    */
                                                  /*                          */
      case FALSE: if (!Fsnext)                    /* "step" wurde bereits ge-*/
                  {  strcpy(buff,dta.name);       /*  rufen; weitersuchen     */
                     return TRUE;                 /*  mit Fsnext.             */
                  }                               /*                          */
                  else                            /*  Nix gefunden; Schalter  */
                  {  first = TRUE,                /*  wieder zuruecksetzen    */
                     return FALSE,                /*  und Fehlschlag signali-*/
                  }                               /*  sieren                  */
      }                                           /*                          */
}                                                 /****************************/
```

Bild 5. TOS-Programmierung unter C: Ausgeben des Directory

lich; GEMDOS hat hier das Vorgehen von MS-DOS übernommen (dem 16-Bit-Betriebssystem für den IBM und Kompatible), das die Atari-Entwickler sehr genau studiert haben...

Wenn Sie das Programm aus Bild 5 mit dem Namen »step« übersetzen, dann können Sie sich zum Beispiel mit dem Aufruf:

```
STEP*.C
```

alle C-Quellen auf der Diskette anzeigen lassen. Aus GEM heraus müssen Sie dazu das Programm zuerst mit der Option »TOS übernimmt Parameter« anmelden. Wenn Sie mehr über die C-Programmierung unter TOS wissen wollen, hilft Ihnen ein in Kürze erscheinendes Buch mit dem Titel »C-Programmierung unter TOS« vom Markt&Technik-Verlag.

Die Hauptattraktion auf dem Atari ist GEM. Der Graphics Environment Manager (wie es mit vollem Namen heißt) kann Fenster (Windows) einrichten, verschieben, wachsen und schrumpfen lassen, beliebige geometrische Objekte zeichnen, die Maus überwachen und vieles andere mehr. Er ist tatsächlich so vielseitig, daß es unmöglich ist, alles in einem Artikel abzuhandeln. Ein anderer Beitrag dieses Heftes geht darauf ein, wie man GEM-Programme in C schreibt und gibt auch einen Eindruck vom Leistungsumfang von GEM wieder.

Sie sehen, daß die Programmierung in »C« kompliziert genug ist.

Aber wenn man einige Punkte beachtet, kann man durch die Benutzung von GEM Programme mit einer sehr effektiven Benutzerführung schreiben. Lassen Sie sich von Anfangsschwierigkeiten nicht abschrecken.

Haben Sie aber erst mal »Ihr« C-System, dann werden Sie feststellen: C-Programmierung macht Spaß!

(Peter Rosenbeck/hb)

Literatur
Brian W. Kernighan, Dennis B. Ritchie: The C Programming Language. Prentice Hall. 1978
Deutsche Übersetzung:
Programmieren in C, Hanser Verlag, ISBN-3-446-13878-1, 48 Mark

---

Zwei C für eine runde Sache

Zwei C-Compiler sind zur Zeit für den Atari ST lieferbar. Jeder hat seine Stärken und Schwächen. Während sich der Compiler von Digital Research für Profis, eignet ist der GST-Compiler auch für Einsteiger geeignet.

Momentan ist das Angebot an C-Compilern noch recht dürftig: außer den beiden von Digital Research und GST waren bei Redaktionsschluß keine weiteren am Markt. Der DRI-Compiler ist ein Profi-System, das zwar noch einige Macken hat (der Compiler und die Bibliothek sind nicht frei von Fehlern!), mit dem sich aber vernünftig arbeiten läßt. Auf ihn werde ich nicht mehr weiter eingehen.

Wegen des begrenzten Angebots ist die Qual der Wahl augenblicklich noch nicht sehr groß. Doch viele Firmen haben eigene Compiler angekündigt, und allzu lange werden diese nicht mehr auf sich warten lassen. Für rosigere Zeiten möchte ich Ihnen ein paar Auswahlkriterien für C-Compiler in die Hand geben.

Als Erstes sollten Sie darauf achten, wie vollständig die Sprache implementiert ist. Wenn alle Schlüsselwörter der Tabelle verfügbar sind, dann ist die Sprache vollständig. Viele Entwickler verzichten jedoch auf das eine oder andere Leistungsmerkmal. Da muß man sich genau überlegen, ob das für die eigenen Zwecke tragbar ist.

So gibt es reine Ganzzahl-Compiler, die keine Fließkomma-Arithmetik kennen (die also auf »float« und »double« verzichten); dazu gehört auch der GST-Compiler. Wollen Sie hauptsächlich nicht-numerische Anwendungen programmieren (Systemprogramme, Textverarbeitung, Datenbanken, die meisten Spiele), dann können Sie dieses Manko in Kauf nehmen. Wer jedoch Geschäftsprogramme erstellt, bei denen mit Geldbeträgen gearbeitet wird, aber zum Beispiel auch Zinsbeträge berechnet werden müssen, der kann mit so einem Produkt nichts anfangen.

Beim GST-Compiler hat man auch auf die Strukturen verzichtet (Schlüsselwort »structure« fehlt). Dies ist eine in meinen Augen bedenkliche Einschränkung. Gute Algorithmen gehen immer einher mit guten Datenstrukturen und dafür ist man auf die Structures angewiesen. Für Profi-Programmierer wird der GST-Compiler aus diesem Grund wohl nicht in die engere Wahl kommen.

Dafür bietet der GST-Kandidat sehr bequeme Bibliotheksfunktionen zur GEM-Programmierung; in diesem System ist die Arbeit mit GEM lange nicht so umständlich wie mit dem C von DRI! Damit wären wir auch schon beim nächsten Beurteilungskriterium, nämlich der Güte und dem Umfang der Bibliotheken, ohne die kein C auskommt.

Zur Grundausstattung gehört eine Standard-Bibliothek. Über deren Umfang kann man sich informieren, indem man die Sprachbeschreibung von Kernighan und Ritchie, die auch den C-Standard definierte, zu Rate zieht. Dieser sogenannte K&R-Standard sollte in der Bibliothek unter keinen Umständen fehlen.

Dann aber stellt sich die Frage, wie hoch hinauf oder wie weit hinunter Sie mit der Bibliothek kommen, sprich: ob Sie Ihnen bequemen Zugriff zu GEM auf der einen, und bequemen Zugriff zu den Routinen des Betriebssystems auf der anderen Seite erlaubt. Was GEM betrifft, ist der GST-Compiler vorbildlich. Beim Zugriff auf das Betriebssystem schränkt er den Programmierer jedoch ziemlich ein. Die Bibliothek enthält keine der Funktionen »gemdos«, »bios« und »xbios«, sondern lediglich eine kleine Auswahl vordefinierter »gemdos«-Funktionen. Für den Profi ist das kein Problem: Mit ein wenig Assembler-Kenntnissen kann man sich die fehlende Betriebssystem-Schnittstelle selber stricken. Dennoch auch diese Einschränkung sollten Sie bei einer Kaufentscheidung berücksichtigen.

Die letzte Frage befaßt sich mit den Extras: Wird zum System ein Editor hinzugeliefert? GST macht das – sehr lobenswert! Gibt es Utilities zur Fehlersuche? (Unix-Programmierern ist ein Programm mit dem Namen »lint« ein Begriff, das viele beliebte Anfänger-Fehler erkennt). Gibt es vielleicht gar eine Ablaufüberwachung (Trace)?

Haben Sie alle für Sie wichtige Punkte beachtet, dann lassen Sie sich von der Leistungsfähigkeit Ihres C-Compilers überraschen.

(Peter Rosenbeck/hb)

Info: C-Compiler von Digital Research Inc. Vertrieb ausschließlich über Atari. Nur erhältlich mit dem Entwicklungspaket für den Atari ST, Preis 969 Mark
C-Compiler von GST. Ob der Vertrieb über Atari oder Softline erfolgt, war bei Redaktionsschluß noch nicht entschieden. Preis zirka 300 Mark.

PROGRAMM-SERVICE

Bestellungen in der Schweiz: Markt&Technik Vertriebs AG, Kollerstr. 3, CH-6300 Zug, Tel. 042/415656
Bestellungen in Österreich: Ueberreuter Media Handels- und Verlagsges. mbH, Alser Str. 24, 1091 Wien, Tel. 0222/481538-0
Bestellungen aus anderen Ländern bitte per Auslandspostanweisung!

Das Angebot dieser Ausgabe:

DUEX ST
Diskettenmonitor mit vielen Funktionen für den Atari ST (Veröffentlicht in Happy-Computer Heft 2/86)

Apfelmännchen
Das berühmte Apfelmännchen als Modula-2-Programm

Puzzle
Ein Desk-Accessoire als C-Listing. Dieses Programmbeispiel führt in die C-Programmierung unter GEM ein.

Plot/Line
Eine Plot- und Line-Routine in Assembler programmiert. Nur auf dem Farbbildschirm ablauffähig. Durch die sehr ausführliche Dokumentation läßt es sich leicht an individuelle Bedürfnisse anpassen.

Maus-Editor
Dieses Basic-Programm verwendet viele GEM-Routinen. Damit läßt sich der Maus-Cursor für Basic-Programme nach individuellen Vorstellungen ändern.

Bagger-Programm
Mit diesem Modula-2-Programm lassen sich Daten zwischen dem Atari ST und anderen Computern übertragen.

Alle Programme liegen in dem jeweiligen Source-Code und als ablauffähige Version vor.
Diskette 3½" für ATARI ST
Bestell-Nr. LH 86S3 D — **DM 29,90*/sFr. 24,90/öS 299,–**

Programme aus früheren Ausgaben:

Ausgabe 3/86
Commodore 64/Commodore 128
Copter-Fight
Ein interessantes Hubschrauber-Kampfspiel für zwei Personen, Geschicklichkeit und Reaktionsvermögen sind Trumpf.
Husky-Basic
Die mächtige Basic-Erweiterung für Grafik, Sound und strukturiertes Programmieren.
Unser Sonnensystem
Lernen Sie mit diesem Anwendungs-Programm die Planeten unseres Sonnensystems kennen (mit Simons Basic)
Wahlautomat
Hardware-Bastelei. Lassen Sie Ihren C 64 Telefonverbindungen anwählen! Zusatz zum Listing des Monats aus HAPPY 2/86
Softpaint
Ein menügesteuertes Zeichen- und Malprogramm für den Commodore 128 im C 128-Modus (kein C 64-Programm).
Bestell-Nr. LH 8603 CD
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 2/86
Commodore 64
Dream Pattern
Machen Sie die Kurvendiskussion auf dem C 64 interessant und nutzen Sie gleichzeitig die tollen Grafikmöglichkeiten dieses Computers voll aus
Börse
Lernen Sie das Börsengeschehen spielend kennen. »Börse« simuliert mit Grafik und Text die Abläufe und Vorgänge an der Börse.
Poster Hardcopy
Dieses Programm fertigt auf Ihrem Drucker einen 75 x 56 cm großen Ausdruck des Commodore-64-Grafik-Speicher an
Kassetten-Designer
Eine hervorragende Hilfe bei der Archivierung von Ihren Computer- oder Musikkassetten
Super-Sprite
Eine Maschinencode-Routine zur professionellen Sprite-Bewegung. Machen Sie Ihren Commodore zu einem Trickfilm-Generator
Transbit
Das Listing des Monats ist ein Terminalprogramm der Spitzenklasse für Ihren Commodore 64. Datenfernübertragung ist kein Problem mehr
Alle 6 Programme auf Diskette für den Commodore 64.
Bestell-Nr. LH 8602 CD
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 1/86
Commodore 64/Commodore 128
Tax. Aus Ausgabe 1/86
Musik und Farbe. Aus Ausgabe 12/85
SDB-Sprite Mover. Aus Ausgabe 1/86
ES-AE. Aus Ausgabe 1/86
Ultraload. Aus Ausgabe 1/86
Error 64. Aus Ausgabe 1/86
Scroll 64. Aus Ausgabe 1/86
Schatzsuche. Aus Ausgabe 12/85
S.L.A.D. Aus Ausgabe 12/84
Alle 9 Programme auf Diskette für den Commodore 64/128
Bestell-Nr. LH 8601 CD
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 12/85
Atari 800XL/130XE/800
Turbo-Basic, auf der Diskette befindet sich je eine Version für den Atari 800XL und eine für den Atari 800 mit mindestens 48 KByte-RAM. AMPEL, Atari-Prüfsummer, Jumper II, Magic Painter
Alle 5 Programme auf Diskette für den ATARI 800XL/130XE/800
Bestell-Nr. LH 8512B
DM 29,90*/sFr. 24,90/öS 299,–

Schneider CPC
Ausgabe 10/85
Programmtransfer leicht gemacht (zwei Programme, S. 72).
»Tasword 464« mit DIN-Tastatur
Bewegte Grafik mit drei Befehlen
Maschinencode-Routinen in BASIC umgesetzt.
Aus Ausgabe 10/85.
Sam – (fünf Programme, S. 109). Aus Ausgabe 11/85
Deutscher Zeichensatz unter CP/M. Hardcopy RSX-Befehle mit direkter Stringvariable. Aus Ausgabe 12/85
Alle 8 Programme auf einer Kassette oder Diskette für den Schneider CPC.
Bestell-Nr. LH 8512 G (Kassette)
DM 29,90*/sFr. 24,90/öS 299,–
Bestell-Nr. LH 8512 D (Diskette)
DM 34,90*/sFr. 29,50/öS 349,–

Ausgabe 11/85
Commodore 64
Aquantor, Zykloide, Nebenkostenabrechnung, Neuer Checksummer, Plakat, Data-Zeilen-Wandler, Super-Saver, More Memory, Autolistschutz, Grafik Window-Zeichner
Aus Ausgabe 10/85
Flugplanung, Finanzen 64, User-Port-Anzeige, Amadeus, ZX81-Utility, Long-Screen, Chess-Screen, Colour-Screen, Autobeep.
Aus Ausgabe 11/85.
Alle 19 Programme auf einer doppelseitig bespielten Diskette für den Commodore 64
Bestell-Nr. LH 8511 A
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 10/85
Sinclair Spectrum
Das »andere« Grafikprogramm. Aus Ausgabe 7/85
Mini-Textverarbeitung. Aus Ausgabe 8/85
Terminal-Programm. Aus Ausgabe 9/85
Alle 3 Programme auf Kassette für den Sinclair Spectrum.
Bestell-Nr. LH 8510 D
DM 19,90*/sFr. 17,00/öS 199,–

Atari 800XL
Prüfsummer, Geröllheimer. Aus Ausgabe 5/85
24 Farben in Grafikstufe 0. Aus Ausgabe 6/85
Diskhelp. Aus Ausgabe 8/85
Ölsuche. Aus Ausgabe 8/85
Autostart. Aus Ausgabe 9/85.
Dudu 4.0. Aus Ausgabe 10/85
Alle 7 Programme auf Diskette für den Atari 800 XL.
Bestell-Nr. LH 8510 B
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 9/85
Commodore 64
Bestell-Nr. LH 8509 A (Diskette)
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 8/85
Schneider CPC 464
Bestell-Nr. LH 8508 G (Kassette)
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 7/85
Commodore 64
Bestell-Nr. LH 8507 A (Diskette)
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 6/85
Commodore 64
Bestell-Nr. LH 8506 A (Diskette)
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 5/85
Schneider CPC 464
Bestell-Nr. LH 8505 G (Kassette)
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 4/85
Commodore 64
Bestell-Nr. LH 8504 A (Diskette)
DM 29,90*/sFr. 24,90/öS 299,–

Ausgabe 3/85
Schneider CPC 464
Bestell-Nr. LH 8503 G (Kassette)
DM 29,90*/sFr. 24,90/öS 299,–

Sonderheft 3/86: 68000
Bestell-Nr. LH 86S3 D (2 Disketten)
DM 34,90*/sFr. 29,50/öS 349,–

Sonderheft 2/86: ATARI
Bestell-Nr. LH 86S2 D (2 Disketten)
DM 34,90*/sFr. 29,50/öS 349,–

Sonderheft 1/86: Schneider
Bestell-Nr. LH 86S1 D (Diskette)
DM 34,90*/sFr. 29,50/öS 349,–
Bestell-Nr. LH 86S1 K (Kassette)
DM 29,90*/sFr. 24,90/öS 299,–

Sonderheft 2/85: Schneider
Bestell-Nr. LH 85S2 D (3"-Diskette)
DM 34,90*/sFr. 29,50/öS 349,–
Bestell-Nr. LH 85S2 V (5¼"-Diskette)
DM 34,90*/sFr. 29,50/öS 349,–
Bestell-Nr. LH 85S2 K (Kassette)
DM 29,90*/sFr. 24,90/öS 299,–

Sonderheft 1/85: Spectrum
Bestell-Nr. LH 85S1 D (Kassette)
DM 19,90*/sFr. 17,00/öS 199,–

* Inkl. MwSt. Unverbindliche Preisempfehlung

Bitte verwenden Sie für Ihre Bestellung und Überweisung die eingeheftete Postgiro-Zahlkarte, oder senden Sie uns einen Verrechnungs-Scheck mit Ihrer Bestellung. Sie erleichtern uns die Auftragsabwicklung, und dafür berechnen wir Ihnen keine Versandkosten.

Atari ST im Hardwarerausch

Nicht nur der Computer ist neu. Auch die Peripheriegeräte bieten Überraschungen!

Bereits beim Kauf des Atari ST muß man bezüglich der Peripherie gewisse Vorentscheidungen treffen. Wählt man eine ST-Version mit 512 KByte RAM, so ist man frei in der Wahl zusätzlicher Geräte, wie beispielsweise des Monitors oder des Diskettenlaufwerks. Schafft man sich jedoch den 520 ST+ an, so kommt man neben der Konsole weder um die Maus, noch um ein Diskettenlaufwerk und den monochromen Monitor herum.

Guckloch

Vor allem die Wahl des richtigen Monitors will gut überlegt sein. Im Gegensatz zu den Gepflogenheiten bei Heimcomputern, bestimmt hier die Art des Monitors nicht nur die Zahl der darstellbaren Farben, sondern auch den Grad der Bildschirmauflösung. Besitzt man einen Farbmonitor, so kann man nur in der niederen und mittleren Bildschirmauflösung arbeiten. Die höchste Auflösungsstufe bleibt dem Benutzer – hardwarebedingt – verschlossen. Verwendet man hingegen einen monochromen Monitor, so kennt dieser nur die höchste Auflösungsstufe. Das bringt aber Probleme. Einerseits kann man nicht jedes Programm, das für den Atari ST geschrieben wurde, tatsächlich benutzen. Ein Paradebeispiel ist das Malprogramm »NEOCHROME«, das nur mit Farbmonitor läuft. So sollte man bei der Entwicklung eigener Programme darauf achten, diese möglichst auf beiden Versionen lauffähig zu gestalten. Dadurch wird man wiederum bei manchen Programmen dazu gezwungen, entweder in der Wahl der Farben oder in der gleichzeitigen Darstellung viele Abstriche zu machen.

Ideal wäre also die Anschaffung zweier Monitore, eines farbigen und eines monochromen. Das ist jedoch finanziell gesehen ein Unding. Deshalb kommt man in der Regel nicht umhin, sich für einen zu entscheiden. In den meisten Anwendungsgebieten bringt der monochrome Monitor mehr Vorteile: Durch ein gestochen scharfes Bild lassen sich Buchstaben und Zahlen auch auf längere Zeit ermüdungsfrei lesen. Dies ist vor allem bei der Textverarbeitung und anderen Anwendungsprogrammen ein großes Plus.

Als Alternative zu einem Monitor kann man auch ein Fernsehgerät mit Scart-Eingang an den ST anschließen. Dabei handelt es sich um einen Monitoreingang, der über das RGB-Analogsignal angesteuert wird. Es ist also ausschließlich eine farbige Darstellung möglich. Wer aber bereits ein Fernsehgerät mit Scart-Anschluß besitzt, kommt auf billige Weise immerhin zu einem Ausgabemedium. Ein passendes Anschlußkabel liefert Atari beim 260 ST übrigens gleich mit.

Atari bietet für den ST derzeit zwei verschiedene Diskettenlaufwerke an und zwar die Typen SF 354 und SF 314. Sie können mit über 360 beziehungsweise 720 KByte Speicherkapazität pro Diskette aufwarten. Hält man sich jedoch die Kapazität des Atari 520 ST+ mit 1 MByte RAM vor Augen, so ist die Anschaffung eines Laufwerks mit nur 360 KByte nicht besonders sinnvoll. Auch wenn man sich nur den Atari 260 ST mit 512 KByte RAM zulegen will, sollte man bedenken, daß sich dieser jederzeit auf 1 MByte nachrüsten läßt (siehe Happy Computer, Ausgabe 1/86). Im Zweifelsfall kann man dann also nur rund ein Drittel des theoretisch belegbaren Speicherinhalts auf einer Diskette unterbringen. Zudem handelt es sich auch beim Umfang eines Programmes für den ST um andere Dimensionen, als man es von Heimcomputern her gewohnt ist. Selbst kleinere Programme, beispielsweise in C oder Modula 2, erreichen durch das Einbinden von GEM-Routinen schnell eine Größe von 50 KByte und mehr. Allein die momentane Version des Basic-Interpreters benötigt bereits rund 150 KByte. Ein extremes Beispiel ist ein Programm, mit dem man Bilder eines Wettersatelliten weiterverarbeiten kann. Jedes dieser Bilder belegt die stattliche Anzahl von 320 KByte auf Diskette! Es ist wohl leicht einzusehen, daß in diesem Fall ein 360-KByte-Laufwerk nur wenig Sinn hätte. Bedenkt man, daß eine doppelseitig formatierte Diskette nur wenig mehr als eine einseitige kostet, deren Preis aber bei rund 10 Mark liegt, läßt sich leicht ausrechnen, daß sich der Preisunterschied zwischen den beiden Laufwerken (er liegt bei rund 100 Mark) bereits durch den Kauf von 10 Disketten amortisiert hat. Mit steigender Diskettenzahl kommt man also mit dem Laufwerk SF 314 trotz höheren Anschaffungspreises billiger weg.

Große Auswahl an Diskettenlaufwerken

Da sich Atari dazu entschloß, die Diskettenschnittstelle Shugard-kompatibel auszulegen, ist man in der Wahl seines Diskettenlaufwerks nicht nur auf die Atari-Modelle angewiesen. Ohne Probleme lassen sich auch Diskettenlaufwerke anderer Hersteller an den Atari ST anschließen. Die Controller sind bereits im ST eingebaut, so daß auch die Diskettenverwaltung keinerlei Probleme mit sich bringt. Da aber nur zwei Controller vorhanden sind, kann man auch nur maximal zwei Diskettenlaufwerke betreiben. Die Verwendung von Laufwerken anderer Hersteller bringt mehrere Vorteile. Zum einen sind solche Laufwerke in der Regel bei gleichen Leistungsmerkmalen billiger als die Originallaufwerke. Außerdem kann man statt der von Atari verwendeten 3½-Zoll-Laufwerke auf diese Weise auch 5¼-Zoll-Diskettenlaufwerke benutzen. Dies lohnt sich vor allem dann, wenn man in Erwägung zieht, CP/M-Software zu benutzen, die noch vorwiegend auf diesem Format vertrieben wird. Allerdings ist damit zu rechnen, daß sich das in absehbarer Zeit ändern wird und man diese Software auch im 3½-Zoll-Format erhält.

Bei der Wahl eines passenden Druckers für den ST stehen wohl zuerst ein FX-80 oder FX-80-kompatibler Drucker zur Debatte. Solch ein Drucker wird auch die geringsten Probleme bei der Software-Anpassung bereiten. Ein grafikfähiger Drucker lohnt sich, weil der ST dazu prädestiniert ist, zu jeder Zeit problemlos eine Hardcopy des momentanen Bildschirminhalts anzufertigen. Dazu verwendet man die Desktop-Option »Bildschirm drucken« oder man betätigt einfach die Tasten »Alternate« und »Help« gleichzeitig.

Grundsätzlich erlaubt der Atari ST jedoch den Anschluß jedes beliebigen Druckers. Dabei spielt es keine Rolle, ob es sich um einen Matrix- oder Typen-

raddrucker handelt, oder ob er mit einer seriellen oder parallelen Schnittstelle angesteuert werden muß. Das läßt sich nämlich in einem Desktop-Menü einstellen, so daß zumindest Texte auf allen Druckern problemlos wiedergegeben werden. Auch die Anfertigung eines Druckerkabels fällt nicht schwer, da im mitgelieferten Handbuch die Anschlußbelegung beider Schnittstellen abgedruckt ist.

Wann und zu welchem Preis eine Festplattenstation für den ST in Deutschland erhältlich sein wird, ist derzeit nur schwer zu sagen. Bei Atari spricht man von einer 20-MByte-Festplatte zum Preis von nur rund 700 Dollar. Sie soll demnächst in Amerika ausgeliefert werden. Der Grund für diesen Unsicherheitsfaktor im Erscheinungstermin liegt vor allem darin, daß Atari die Festplatten nicht selbst, sondern bei einem anderen Hersteller in Auftragsarbeit herstellen läßt. Eine Firma zu finden, die zu solch niedrigen Preisen produziert, wie sie der Atari-Geschäftsführung vorschweben, ist aber nicht einfach.

Dadurch ist es auch möglich, daß andere Hersteller früher mit Festplattenlaufwerken auf dem Markt sein werden als Atari, da bereits verschiedene Firmen mit der Entwicklung beschäftigt sind.

Lexikon im Westentaschenformat

Ähnliche Probleme hat Atari mit dem seit langem versprochenen CD-ROM, das eine fantastische Speicherleistung auszeichnet. Auf einer Laserdisk lassen sich unglaubliche Datenmengen speichern, wobei die Zugriffszeiten Festplattenlaufwerke weit in den Schatten stellen. Es gibt bereits ein CD-Laufwerk für den IBM-PC und Kompatible. Ein amerikanischer Verlag stellte eine Laserdisk vor, die eine normalerweise 20-bändige Enzyklopädie beinhaltet. Da sich Laserdisks aber nur lesen und nicht beschreiben lassen, ist der Anwendungsbereich doch etwas begrenzt. Aus diesem Grund will sich Atari mit seinem CD-Laufwerk auch etwas Besonderes einfallen lassen. Neben der Funktion als Massenspeicher soll dieses Laufwerk auch zur Wiedergabe von normalen CD-Schallplatten geeignet sein. Es bleibt abzuwarten, wie diese Vorstellung in Realität aussehen wird.

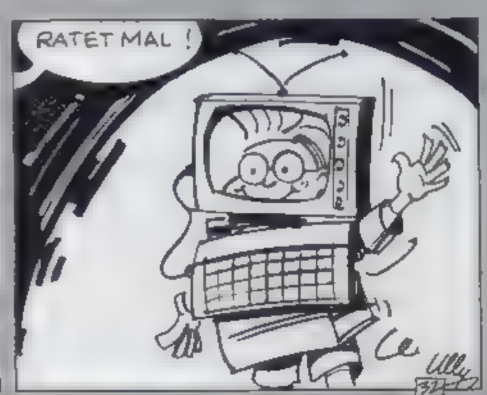
Orchesterklänge mit dem ST

Der Atari ST ist aber nicht nur für Computer-Interessierte, sondern auch für Musiker interessant. Dies liegt jedoch weniger im Sound-Chip des ST begründet, sondern vielmehr an der eingebauten MIDI-Schnittstelle. »MIDI« ist die Abkürzung für Musical Instrument Digital Interface. Es handelt sich hier um eine Normschnittstelle, ähnlich der Schnittstellennormen RS232 und Centronics, auf die sich die Hersteller geeinigt haben. Der Datentransfer erfolgt seriell mit einer Übertragungsrate von 31250 Baud. Dabei werden Befehle und Daten zwischen verschiedenen elektronischen Musikinstrumenten oder einem Computer und einem Instrument ausgetauscht.

Im einfachsten Fall kann man mit Hilfe von MIDI, also beispielsweise durch einen Synthesizer, einen zweiten Synthesizer steuern, so daß dieser zwar dieselbe Melodie, jedoch mit einem anderen Klang spielt. Schließt man jedoch einen Computer an, so erweitert sich die musikalische Bandbreite erheblich. Man kann zum Beispiel die Klangdaten eines Sythesizers speichern, verändern und bei Bedarf wieder abrufen. Viel interessanter ist jedoch die Steuerung und Koordination von mehreren Instrumenten mit Hilfe des Computers. Mittels geeigneter Software kann man ganze Melodien speichern und am Bildschirm editieren. Diese Melodien sind auf Knopfdruck abrufbar, wobei nun bis zu 16 verschiedene Melodien auf bis zu 16 verschiedenen Instrumenten umgesetzt werden können. Ein einziger Computer simuliert also ein ganzes Orchester. Der »elektronische Dirigent« ist geboren! Dies ist vor allem beim Komponieren neuer Melodien von Vorteil, da man keine komplette Kapelle benötigt, um einen Gesamteindruck des fertigen Stückes zu erhalten.

All dies ist natürlich bereits auf anderen Computern realisiert worden. Vor allem auf dem Commodore 64, für den es zur Zeit die meiste Software für diese Anwendungen gibt. Dem Atari ST sollte es jedoch ein Leichtes sein, den C64 aus seiner Führungsrolle zu verdrängen. Zum einen ist die MIDI-Schnittstelle bereits eingebaut, zum anderen ist seine Leistungsfähigkeit höher. Der entscheidende Vorteil gegenüber dem C64 liegt jedoch in der wesentlich höheren Speicherkapazität. Die Klangdaten zur Steuerung von anderen Instrumenten sind nämlich sehr speicherplatzintensiv, so daß man beim C64 schnell an die Grenzen des verfügbaren Speicherplatzes stößt. Zudem verspricht man sich vom ST eine wesentlich übersichtlichere und benutzerfreundlichere Bedienerführung in den Programmen, die durch die höhere Grafikauflösung und durch GEM-Unterstützung leicht zu realisieren sind. Entsprechend hoch ist bereits jetzt das Interesse von Musikern an diesem Computer. (Wolfgang Czerny/hb)

kosinus von GUBA & ULLY



Drucker für die neuen Computer – das paßt

Bild 1. Der LQ-800 – klassisch-schön und leistungsfähig

Passend zur Leistung der neuen Computergeneration in Geschwindigkeit, Grafik, Text- und Datenverwaltung stellen wir Ihnen eine Reihe von Druckern vor, mit denen Ihr Computer erst so richtig gut zum Ausdruck kommt.

Welcher Drucker ist wohl der der richtige? Die Frage läßt sich sicherlich nicht mit einem Satz beantworten. Zu verschieden sind die Wünsche, Ansprüche, Aufgaben und letztendlich auch die Geldbeutel. Trotzdem lassen sich Empfehlungen geben, denn wir haben eine Reihe von Druckern an den neuen Computern, vor allem aber am Atari ST und am Amiga ausprobiert.

Die Anforderungen, die dabei an die Drucker gestellt wurden, liegen auf der Hand. Zum einen sollen sie natürlich den Grafikfähigkeiten der neuen Computer entsprechen, zum anderen ist eine flotte Schönschrift für die Textverarbeitung mehr als wünschenswert. Daß die Drucker dabei noch leicht zu bedienen, solide aufgebaut sein und vielleicht noch die eine oder andere Besonderheit aufweisen sollen, haben wir ebenfalls berücksichtigt.

Schnittstellenvielfalt

Endlich möchte man sagen, endlich haben sich die Computerhersteller dazu durchgerungen, ihre Computer mit der mittlerweile zum Standard gewordenen Centronics-Schnittstelle auszurüsten.

Sowohl der Atari ST als auch der Commodore Amiga verfügen über die genormte Centronics-Schnittstelle und besitzen erfreulicherweise auf der Geräterückseite eine handelsübliche Buchse (Cannon D-Subminiatur), die jedem Drucker mit dieser Norm entspricht. Nur beim Apple Macintosh gestaltet sich das Ganze etwas komplizierter, denn an seiner Schnittstelle lassen sich eigentlich nur Apple-Drucker (Imagewriter) sinnvoll anschließen, deshalb wollen wir Ihnen auch keine weiteren Drucker für den Apple vorstellen – die Auswahl ist zu klein. Anders bei Atari und Commodore, bei diesen Computern hat man prinzipiell die große Auswahl und die Grenze setzt letztendlich nur das zur Verfügung stehende Budget. Wer will, kann sich aber durchaus auch für einen Laser-Drucker oder einen Plotter für 100000 Mark entschließen. Aber zurück zur Realität, denn in der Regel sollte ein Drucker nicht viel mehr als 2500 Mark im Höchstfall kosten. Dafür erhält man heute schon Leistungen, die für fast alle Zwecke ausreichen. Es geht natürlich auch billiger. Wer gewisse Abstriche an der massiven Verarbeitung, manchen Sonderfunktionen und der Geschwindigkeit macht, wird schon unter 1000 Mark gut bedient. Damit neben dem leichten Anschluß auch beim Drucken die Freude nicht verlorengeht, haben sich die Konstrukteure der neuen Computer-Generation etwas einfallen lassen. Ohne besondere Maßnahmen senden die Computer Befehle nach der neuen ESC/P-Norm, die von Epson entwickelt wurde und nun auch den bisherigen Industriestandard in Worte und Zahlen gefaßt hat. Praktisch betrachtet bedeutet das, daß alle Drucker, die wie der FX-85 angesteuert werden, sich ohne Umschweife verwenden lassen. Die anderen Drucker benötigen die Installation eines sogenannten »Druckertreibers«. Beim Amiga ist das extrem einfach, denn man wählt im Kontrollfeld einfach den Namen des bestimmten Druckers aus. Die Liste der Druckertreiber ist dabei eigentlich nur von der Kapazität der Diskette abhängig. Die aber reicht für mehr als 100 Druckertreiber aus. Der Atari ST macht mehr Umstände, da diese Druckertreiber extra zu laden, beziehungsweise in den verwendeten Programmen einzustellen sind

Wir werden Ihnen nun eine Reihe von Druckern verschiedener Hersteller vorführen und dabei auf deren Besonderheiten eingehen. Den Abschluß jedes Abschnitts bildet eine Kurzbewertung des Druckers. In einer Tabelle finden Sie auch nochmals alle Daten der Drucker auf einen Blick zusammengefaßt.

Epson LQ-800, ein Meister seines Faches

Attribute wie »schön, leistungsstark, faszinierend und etwas avantgardistisch« beschreiben den Epson LQ-800 (Bild 1) sicherlich nicht falsch. Vergleicht man sein Schriftbild mit dem der meisten Konkurrenten, so sind die »anderen« eben nur »Drucker« während der LQ-800 sich in die höheren Sphären eines kleinen Wunderwerkes erhebt. So ein »Wunder« hat allerdings auch seinen Preis. Mit 2498 Mark ist der LQ-800 der teuerste Drucker unserer Parade. Obwohl sicherlich nicht billig, paßt der LQ-800 gut zur neuen Computergeneration, denn er läßt sich, wie der bekannte FX-85, von den meisten Textprogrammen und natürlich auch per Maschinen- beziehungsweise Basic-Programm steuern. Alle Programme für den FX-85 (früher FX-80) kann man auf dem LQ-800 ebenfalls verwenden. Dabei wird allerdings etwas

Merkmal / Drucker	Preis	Schriftarten							Grafik	Nadeln	Tasten	Hoch/Tief	Unterstreichen	Pufferspeicher	Geschwindigkeit	
		Pica	Elite	Schmal	Fett	Doppelt	Breit	NLQ							Norm	NLQ
Epson LQ-800	2498	x	x	x	x	x	x	LQ 17 x 37	bis 2880 Punkte/Z	24	On Line LF FF	x	x	4 KByte	180	LQ 60
RFI DP 165	1498	x	x	x	x	x	x	17 x 17	bis 960 Punkte/Z	9	On Line LF, FF	x	x	2 KByte	169	41
NEC P2	1298	x	x	x	x	x	x	21 x 18	bis 1920 Punkte/Z	18	On Line LF, FF	x	x	3,5 KByte	180	29
Epson FX-85	1848	x	x	x	x	x	x	18 x 18	bis 1920 Punkte/Z	9	On Line LF, FF	x	x	8,2 KByte	160	37
Fujitsu DX 2100	1932	x	x	x	x	x	x	18 x 16	bis 1920 Punkte/Z	9	On Line LF FF	x	x	2 - 18 KByte	220	44
Citizen D 120	998	x	x	x	x	x	x	18 x 16	bis 1920 Punkte/Z	9	On Line LF FF	x	x	4 KByte	120	21

Die Vergleichstabelle stellt die Fähigkeiten der sechs Drucker nebeneinander

von seinen Fähigkeiten verschenkt, denn dann liegen 16 der 24 Drucknadeln brach. Seine volle Text- und Grafikfähigkeiten entfaltet der LQ-800 nämlich erst dann, wenn man jede seiner Nadeln auch wirklich drucken läßt. Dazu aber später mehr.

Rein äußerlich ist der LQ-800 eher ein schlichter, sachlicher Geselle, dem man seine Fähigkeiten nicht von außen ansieht. Gerade dieses »Understatement« ist es aber, was ihn zusätzlich reizvoll macht. Die Ausmaße des Gehäuses sind kleiner als die des FX-85, trotzdem verarbeitet der LQ-800 natürlich Endlos- und Einzelpapier, letzteres allerdings nur dann ordnungsgemäß wenn ein zusätzlicher Traktor (125 Mark) verwendet wird. Auf der Gehäuserückseite befindet sich die Centronics-Schnittstelle und direkt daneben eine sechspolige DIN-Buchse mit RS232C-Belegung. Der LQ-800 besitzt ebenso wie der FX-85 eine Gehäuseklappe am rechten hinteren Eck. Darunter befinden sich aber nicht, wie man zunächst vermutet, die DIL-Schalterreihen (die sind an der Gehäuserückseite), sondern ein Modulsteckplatz mit besonderen Fähigkeiten. Ein einfaches Einstecken von extra erhältlichen Schrift- (Courier, Script, Prestige Elite, Sanserif und OCR B; Preis pro Modul 125 Mark) oder Steuermodulen (IBM, Epson extended, Diabolo 630; Preis pro Modul 198 Mark) verändert das Aussehen des Schriftbildes beziehungsweise die Befehlstabelle des Druckers in beinahe jeder gewünschten Weise. Ein Blick auf den relativ zierlichen Druckkopf mit seinen in zwei versetzten Reihen angeordneten 24 Drucknadeln überzeugt, welche vielfältigen Schriftvariationen denkbar sind. Aber auch ohne Schriftmodule ist schon einiges an Abwechslung geboten. Neben den in der ESC/P-Norm festgelegten Schriftarten wie fett, breit, schmal, unterstrichen, proportional, hochgestellt, tiefgestellt, und doppelt steht ein eigener Pufferspeicher von 7 KByte für eigene Zeichen zur Verfügung. Alle diese Variationen können durch die »Master Style«-Betriebsart auf einfache Weise miteinander kombiniert werden. Die eigentliche Sensation ist aber die »Letter Quality«-Schrift, bei der das »Near« davor getrost wegfallen darf, denn die Punktmatrix hat eine Auflösung von 37 x 17 Punkten. Der Unterschied zu einer Typenradschreibmaschine ist wirklich kaum mehr der Rede wert, allerdings hat man mit dem LQ-800 auch in der Schönschrift wesentlich mehr Schriften zur Auswahl als mit einem Typenrad. Sogar bei der äußerst kritischen Italic-Schrift (Schrägschrift) muß der LQ-800 nicht passen, die Ränder sind sauber und lassen kaum Einzelpunkte erkennen.

Ganz schön flott

Trotz seiner vierundzwanzig Beine kommt der LQ-800 nie ins Stolpern. In der Normalschrift (9 x 17 Punktmatrix), die sich übrigens auch sehen lassen kann, schafft er flotte 180 (gemessen 165) Zeichen pro Sekunde. Davon bleiben im Schönschriftmodus (LQ) volle 60 (gemessen 57) Zeichen pro Sekunde übrig, da der LQ-800 nicht wie 8-Nadel-Drucker mehrfach über eine Zeile streichen muß. Der LQ-800 arbeitet im Schönschrift-Modus weiterhin wie gewohnt, indem er vorwärts und rückwärts einmal pro Zeile druckt, wobei Leerstellen übersprungen werden. Die Geschwindigkeit, mit der das Papier weiterbefördert wird, entspricht ungefähr der des FX-85.

Als wahrer Künstler entpuppt sich der LQ-800 im Grafikbetrieb. Bei den schon vom FX-85 bekannten Grafikmodi mit einer Auflösung von bis zu 1920 Punkten pro Zeile (vierfache Dichte) bemüht der LQ-800 lediglich acht Nadeln. Aktiviert man dann die restlichen Nadeln mit einfachen ESC-Befehlen, wird es beinahe unheimlich. Im höchstauflösenden Modus werden 2880 Punkte pro Zeile gedruckt (sechsfache Dichte).

Nur ein Traum?

Verwendet man in diesem Modus ein frisches Farbband und poröses Papier, kann man das Gedruckte in exzellenter Qualität sowohl auf der Vorderseite als auch auf der Rückseite des Papiers wiederfinden. Hat man aber glattes Papier eingespannt, so ist es problemlos möglich, beispielsweise reproduktionsfähige Platinenlayouts zu drucken.

Seine überlegenen Leistungen, aber auch sein Preis verleihen dem LQ-800 das Image eines Traumdruckers. Aber müssen Träume Illusion bleiben? Wer Wert auf höchste Flexibilität, exzellentes Schriftbild und hohe Geschwindigkeit legt, findet im LQ-800 einen zuverlässigen Partner für viele Jahre

Epson FX-85: Altbewährtes verbessert

Wie bei fast allen Entwicklungen gibt es immer mehrere Wege, ein Produkt an veränderte Marktbedingungen anzupassen. Die erste Möglichkeit besteht darin, ein völlig neues Gehäuse mit neuer Technik zu entwickeln, die zweite Art beläßt an einem Produkt die guten Teile und verbessert nur das nicht mehr Zeitgemäße. Dafür entschloß sich auch Epson. Deshalb unterscheidet sich der FX-85 (Bild 2) rein äußerlich auch nur durch einige wenige, aber wesentliche Merkmale. Außer dem Namensschild ist

das vor allem die neue Bezeichnung der Funktionstasten. Neben dem altbekannten On Line, Form Feed und Line Feed findet man zwei weitere Bezeichnungen; Draft und NLQ. Und tatsächlich, auch in dieser Preisklasse ist bei Epson das Zeitalter der Near-Letter-Quality-Schrift angebrochen. Die Buchstaben »Draft« stehen lediglich dafür, daß man die mit NLQ einstellbare, besonders schöne Schrift auch durch Tastendruck wieder abschalten kann.

Die Mehrzahl der Veränderungen hat intern im Drucker stattgefunden. Mit einer Zusatzplatine und neuen Steuer-ROMs sind einige Funktionen dazugekommen, die das Arbeiten mit dem FX-85 sowohl für den Basic-Programmierer als auch für denjenigen, der mit verschiedenen Text- und Datenverarbeitungsprogrammen arbeiten möchte, angenehm und effektiv gestaltet. Da ist zunächst die sogenannte Master-Betriebsart. Sie erlaubt es, mit einem einzigen Befehl jede nur erdenkliche Schriftenkombination sowohl im Normalschrift-, als auch im Schönschriftmodus zu erreichen. Man sucht sich nur noch aus einer Tabelle in dem exzellenten Handbuch die gewünschte Schrift heraus und gibt die Nummer der Schrift in Form eines Parameters an. Auch für die von vielen Textverarbeitungen her bekannte Funktion des Randausgleichs (rechts, Mitte, links) wird vom FX-85 unterstützt. Neu sind auch die Befehle zur Beeinflussung des achten Bits der ankommenden Daten.

Der FX-85 unterscheidet sich mechanisch nur unwesentlich vom FX-80. Er besitzt ebenso neun Nadeln und druckt die Zeichen der Normalschrift in der bekannten 9x11-Matrix. Schaltet man jedoch in die Schönschrift, so wird die Matrix, wenigstens rechnerisch, zu einer 18x18-Matrix. Dieser Punktevorteil wird durch einen nicht unerheblichen Geschwindigkeitsnachteil (37 gegenüber 160 Zeichen pro Sekunde) erkauft. Im Gegensatz zu seinem Vorfahren besitzt der FX-85 einen Pufferspeicher von acht KByte. Damit erreicht er bereits Dimensionen, ab denen sich ein Pufferspeicher bezahlt macht. Man kann mit der Arbeit fortfahren, während der Drucker noch den letzten Ausdruck beendet.

Drucker kann man auf die verschiedensten Arten steuern; durch Schalter, DIL-Schalterreihen, Sekundäradressen und natürlich mit den ESC-Befehlen. Die Drucker mit dem Namen Epson zeichneten sich immer schon durch ihre flexiblen Befehle, die das gesamte Leistungsspektrum des Druckers auch wirklich verfügbar machen, aus. Beim FX-85 wurde von diesem Konzept glücklicherweise nicht abgewichen. Die Kontrolle über den Drucker ist mit einer enormen Vielzahl an Befehlen möglich. Beim FX-85 ist aber noch eine weitere Steuerungsmethode hinzugekommen. Neben der bereits erwähnten Schönschrift lassen sich mit den Funktionstasten des Druckers neun verschiedene Schriften (siehe Tabelle) und ein Steuerbefehl zum Überspringen der Perforation einstellen. Die Bedienung ist einfacher, als man denkt. Man drückt einfach die On-Line und Form-Feed-Taste gleichzeitig und befindet sich im Programmiermodus. Nun kann man durch mehrmaliges Drücken auf die On-Line-Taste die gewünschte Schrift einstellen. Nach der Bestätigung mit der Form-Feed-Taste und der Rückstellung des Programmiermodus mit der Line-Feed-Taste ist der Drucker in der gewünschten Schrift fixiert. Diese Funktion ist wohlgemerkt nur so lange sinnvoll, wie man alternativ auf die Programmierung der ESC-Befehle zurückgreifen kann.

Mit dem FX-85 ist es Epson gelungen, der Konkurrenz ein Schnippchen zu schlagen. Zwar gibt es schnellere Drucker und auch solche, die ein wenig schöner schreiben. Die gleiche Kombination aus Leistung und Bedienungskomfort kann aber kein anderer Drucker dieser Preisklasse anbieten. Dabei ist es erfreulich, daß der FX-85 keinen Pfennig mehr als der alte FX-80+ kostet – er hat den gleichen Listenpreis von 1848 Mark. Nun werden sich manche Besitzer des FX-80/FX-80+ überlegen, daß man Ihren Drucker vielleicht aufrüsten könnte. Kurz gesagt – man kann! Für beide Drucker gibt es von Epson Umbausätze für 350 Mark (FX-80) beziehungsweise 300 Mark (FX-80+). Der Einbau soll nach Aussage der Firma Epson von jedem autorisierten Fachhändler vorgenommen werden können.

Mit seinen Leistungen wird der FX-85 auch weit über das Jahr 1985 noch tonangebend in der Druckerwelt sein. Und selbst wenn es einmal eine weitere Verbesserung geben sollte, bei der derzeitig von Epson verfolgten Strategie wird man wahrscheinlich einen Umbausatz erhalten können.

Geheimtip: RFI DP 165

Der DP 165 (Bild 3) ist ein Drucker, der ursprünglich für den professionellen Einsatz an Personal Computern wie dem IBM-PC entwickelt wurde. Dementsprechend hoch ist sein Leistungsniveau. Mit mehr als 165 Zeichen pro Sekunde gehört der DP 165 zu den ausgesprochen schnellen Druckern. Trotz der hohen Geschwindigkeit überrascht der DP 165 durch einen angenehm niedrigen Geräuschpegel. Das alleine macht natürlich noch keinen Drucker der Spitzenklasse aus. Der DP 165 bietet eine enorme Vielfalt an Funktionen und Schriftarten. Bei der Syntax der Steuerbefehle glänzt der DP 165 durch vollständige Gleichheit mit dem Epson FX-80. Der Unterschied zum FX-80 liegt in der NLQ-Schrift (Near Letter Quality = Schöndruck), die sich sogar mit proportionalen Zeichenabständen drucken läßt.

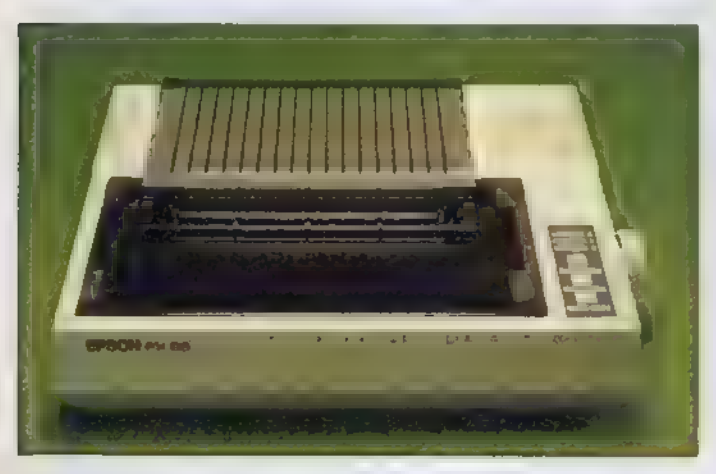


Bild 2. Bewährt wie eh und je – dezenzt in Design, Spitze in der Leistung – der Epson FX-85

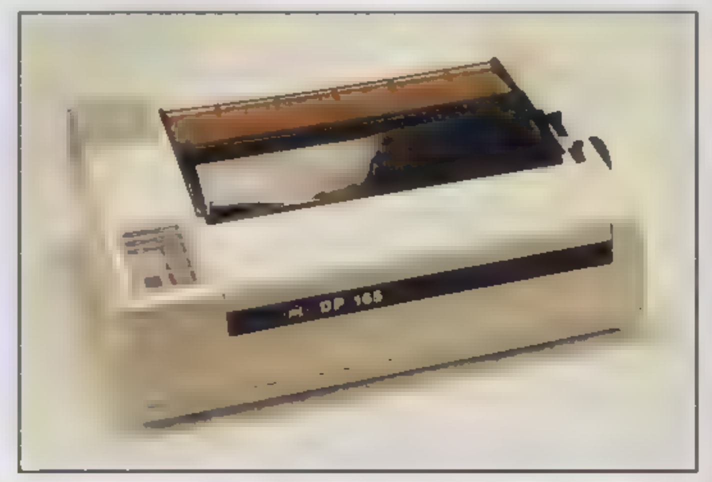


Bild 3. Ein echter Profidrucker – der RFI DP 165

Die NLQ-Schrift ist auf drei verschiedene Arten einzustellen. Einmal durch einen Steuerbefehl, durch eine separate Taste auf dem Bedienfeld und durch einen der gut erreichbaren DIL-Schalter auf der Gehäuserückseite. Die Qualität dieser Schrift kann sich sehen lassen, zumal der DP 165 in der NLQ-Schrift immer noch schneller (und leiser) druckt als die meisten Typenraddrucker. Normalerweise ist das Aussehen der NLQ-Schrift durch die im ROM des Druckers abgelegten Werte bestimmt (17 x 17 Matrix). Erweitert man das interne RAM von 2 KByte auf 8 KByte, wird es sogar möglich, eine eigene NLQ-Schrift zu entwerfen. Ohne die RAM-Erweiterung bleibt der selbstdefinierte Zeichensatz auf die, schon vom FX-80 her bekannte, 9 x 9 Zeichenmatrix beschränkt. Eine andere Methode, selbstdefinierte Zeichen und Grafiken auszudrucken, besteht in der Verwendung der umfangreichen Grafik-Fähigkeiten des DP 165. Alle Hardcopy-Routinen, die für die Epson-Drucker programmiert wurden, funktionieren, ohne jede Änderung, auch zusammen mit dem DP 165.

Der praktische Wert eines Druckers hängt allerdings nicht nur von seinen Fähigkeiten, sondern auch vom Bedienungskomfort ab. Dazu gehört ein einfach zu bedienender Papiereinzug, ein Papiertransport, der ein direktes Abreißen des Papiers über dem Druckkopf ermöglicht, und eine Druckersteuerung über Funktionstasten. Alle diese Bedingungen erfüllt der DP 165. Das Papier, das entweder Einzel- oder Endlospapier mit Randlochung sein kann, wird ohne großen Aufwand von hinten in den Drucker eingelegt und durch zwei Stachelwalzen transportiert. Die Stachelwalzen sind in ihrem Abstand beliebig verstellbar. Auch das Einlegen des Farbbandes ist nicht schwieriger als bei einem FX-80, zumal dessen Farbbänder problemlos passen. Die vier Bedienungstasten des DP 165 haben neben den Standardfunktionen wie Zeilen- und Seitenvorschub noch eine weitere Bedeutung erhalten. Mit ihnen lassen sich auf einfache Weise die linken und rechten Ränder des Schreibfeldes einstellen.

Der DP 165 ist ohne Zweifel ein Drucker der Spitzenklasse. Mit einem Preis von 1498 Mark hält er sich, gemessen an seinen Fähigkeiten, in erfreulich bescheidenen Grenzen. Seine vollständige Übereinstimmung mit den Funktionen der Epson FX-80-Drucker, ergänzt durch die NLQ-Schrift und den niedrigen Geräuschpegel, zeichnen den DP 165 aus. Wer sich etwas abseits von den Pfaden der größeren Druckerhersteller umschaut, kann also durchaus so manche Mark sparen

Der NEC-P2 — das fernöstliche Wunder

Packt man den NEC-P2 (Bild 4) aus, so erscheint ein kompaktes, eher unscheinbares Gerät. Nichts deutete anfangs darauf hin, zu welchem Wunderding er sich im Test entwickeln sollte. Ein Gehäuse mit den Tasten LF, TOF, FONT und SEL an der Vorderseite, ein Friktionsantrieb und ein aufsetzbarer Traktor — nichts Ungewöhnliches könnte man meinen. Aber die Würze liegt im Detail. Schon ein erster Blick in das sehr gute deutsche Handbuch macht neugierig. Die gesamten Möglichkeiten des NEC-P2 sind dort klar, übersichtlich und, was noch wichtiger ist, gut verständlich dargestellt.

Schnell klärt das Handbuch über die Bedeutung der FONT-Taste auf: Mit ihrer Hilfe lassen sich verschiedene Schriftvarianten einstellen. Im einzelnen sind dies drei Varianten des Typs Pica, zwei Versionen der Elite- und ebensoviele der Schmalschrift. Eine 7-Segment-LED-Anzeige auf der Bodenplatte des P2 zeigt den jeweils gewählten Schrifttyp in Form einer Zahl an. Wenn der Druckkopf nicht gerade darüber steht, gibt es auch keine Probleme beim Ablesen. Natürlich sind alle durch Tasten getroffenen Einstellungen auch mittels ESC-Sequenzen zu verändern.

Doch dies ist nur der erste Eindruck! Neben der eben erwähnten Anzeige befinden sich zwei leicht zugängliche 8fach-DIP-Schalter. Sie dienen der Einstellung des Zeilenabstandes, des Perforationssprunges, der Seitenlängen und vielem mehr. Selbst die Vorwahl des Schriftbildes beim Initialisieren (Einschalten) des Druckers ist hier möglich.

Als außergewöhnlich gut gelungen muß die Tatsache angesehen werden, daß es beim NEC-P2 möglich ist, Schnittstellen ganz nach Belieben und ohne Werkzeug auszutauschen. An der Rückseite des NEC-P2 befindet sich eine Klappe, die den Blick auf eine eingeschobene Baugruppe freigibt. Dieses Schnittstellen-Element wird nach oben herausgezogen und ein anderes (zum Beispiel serielle RS232) findet an gleicher Stelle seinen Platz.

Die dem NEC-P2 eigenen elf Zeichensätze befinden sich ebenfalls in den Schnittstellenbaugruppen. Mittels DIP-Schaltern, die natürlich von außen zugänglich sind, wird die Auswahl getroffen. Wird die Papierabdeckung während des Druckes geöffnet, so bleibt der NEC-P2 augenblicklich stehen. Aus der dann entstehenden Stille zieht ein leises, unaufdringliches Geräusch die Aufmerksamkeit auf sich Wer sagt da, im Lande der aufgehenden Sonne werde nur billigst produziert? Der NEC-P2 ist der Gegenbeweis. Das Geräusch entstammt einem Lüfter, der im Gehäuse für das notwendige Klima sorgt.

Zur Geräuschminderung hat man ihm eine automatische Abschaltung des Lüfters spendiert. Diese ist wirksam, wenn der NEC-P2 länger als zirka 30 Sekunden keine Daten vom Computer empfängt. Zum Schutz des Druckkopfes hat man ihn ebenfalls mit einem Thermofühler ausgestattet, der, sollte es ihm zu warm werden, erst einmal auf unidirektionalen Betrieb umschaltet. Reicht dies immer noch nicht aus, um abzukühlen, so verordnet diese Schutzeinrichtung eine Zwangspause.

War das Äußere schon beeindruckend, so sollte sich der NEC-P2 natürlich auch im harten Alltag bewähren. Im reinen Geschwindigkeitstest (10 Zeilen) ermittelten wir eine durchschnittliche Geschwindigkeit von 160 Zeichen/Sekunde. Damit bleibt der P2 etwas unter den angegebenen 180 Zeichen/Sekunde (Tabelle). Auch wenn dieser Wert in der NLQ-Schrift auf 29

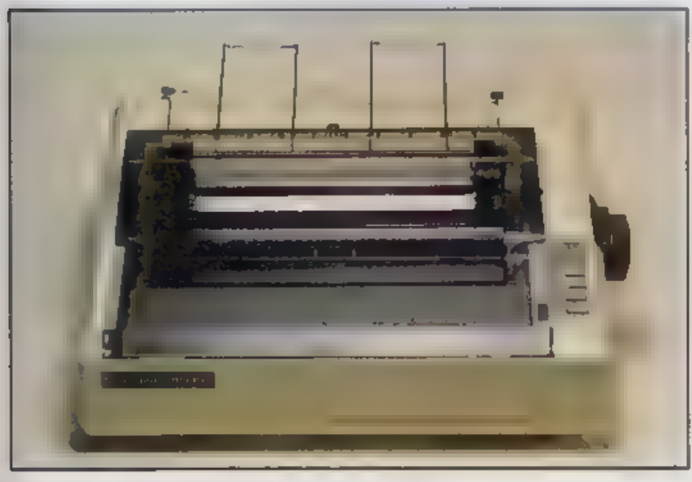


Bild 4. Der NEC-P2 mit aufgesetztem Traktor

Zeichen/Sekunde zurückgeht, so ist dies immer noch ein sehr guter Wert.

Der NEC-P2 hat einen gänzlich anderen Befehlssatz als unser Referenzdrucker, der FX-85. Aber jedes gute Textverarbeitungsprogramm läßt das Übermitteln von druckerspezifischen ESC-Befehlen zu. Damit sind dem Besitzer des P2 Tür und Tor zu den reichhaltigen Möglichkeiten des Druckers geöffnet. Die Vielfalt der Schriftbilder und Druckmodi wird durch reichliche Mischmöglichkeiten ergänzt. Je nach Druckmodus arbeitet dieser Drucker mit einer 7x9-(Schmalschrift-) bis hin zu einer 21x18-(Pica-NLQ-) Matrix. Natürlich darf nicht vergessen werden, daß den NEC-P2 Einzelnadelansteuerung nicht ins Schwitzen bringt. Im 8- oder 16-Bit-Modus nadelt er alles aufs Papier, was er an Grafik angeboten bekommt.

Der NEC-P2 bestätigte bis zum Ende des Tests den guten Eindruck der ersten Stunde. Der für den Dauerbetrieb so wichtige mechanische Aufbau konnte überzeugen. Der Druckkopf wird mittels zwei Laufschienen geführt, was der Präzision und damit dem Schriftbild zugute kommt. Als Antrieb dient ein Zahnriemen, der Kopf selbst ist gegen Verschmutzen gut geschützt und großzügig dimensioniert. Alles Voraussetzungen, die ein hohes Maß an Zuverlässigkeit erwarten lassen.

Ein professioneller Drucker für den entsprechenden Anwenderkreis. Damit das gute Gesamtbild auch komplett ist, gibt es zum NEC-P2 einen Einzelblatteinzug, der auch funktioniert. Insgesamt kann zwischen Einzelblatteinzug, Stachelradantrieb (165 Mark), bidirektionalem Formulartraktor (456 Mark) und der genannten automatischen Einzelblattzuführung gewählt werden.

Der P2 ist ein Drucker, der die Konkurrenz zum Handeln zwingt. Kompromißlos in Schrift und Ausstattung, ist er ohne Zweifel eine gute Empfehlung. Mit einem Preis von 1812 Mark ist er darüber hinaus eine Herausforderung für die Konkurrenz.

Fujitsu DX 2100 – zwei zum Preis von einem

Der DX 2100 (Bild 5) strengt sich an, zwei Drucker in sich zu vereinigen, allerdings mit der Absicht, beide in ihren Leistungen zu übertreffen. Erreicht wird diese Flexibilität durch einen sinnvoll konstruierten Farb-Umrüstsatz, der so einfach wie schnell einzubauen ist. Hat man diesen Umbausatz installiert, so genügt es, zwischen schwarzer und vierfarbiger Farbbandkassette, die sich garantiert ohne schmutzige Finger einlegen lassen, zu wählen, um den betreffenden Modus zu aktivieren. Eine Kerbe auf der Kassette signalisiert dem Drucker, ob er die Erlaubnis hat, das Farbband vor dem Kopf auf- und abzuschieben, um eine andere Farbe einzustellen. Die Beschränkung auf vier Farben des Farbbandes bedeuten, für den DX 2100 noch lange nicht, daß er nicht auch mehr hervorbringen kann. Die additive Farblogik sorgt dafür, daß Mischfarben nicht überproportional betont werden. So wie die Farbfähigkeit als sinnvolle Ergänzung zu betrachten ist, so bleiben die monochromen Fähigkeiten eines Druckers bei weitem die wichtigeren Beurteilungskriterien. Das Schriftbild ist hier entscheidend. Um aber ein gutes Schriftbild überhaupt und auf Dauer erzeugen zu können, bedarf es einer Menge Stahl. Beim DX2100 hat dieser Stahl die Form von zwei soliden Gleitschienen angenommen, auf denen der vollkommen gekapselte 9-Nadel-Druckkopf mit der Präzision einer Schweizer Uhr gleitet. Eigentlich ist gleiten nicht ganz das richtige Wort, denn die Geschwindigkeiten, die der Druckkopf des DX 2100 entwickelt, sind beeindruckend. Mit einer Geschwindigkeit von angegebenen 220 Zeichen pro Sekunde (gemessen 210) ist er einer der schnellsten Drucker. Der eigentliche Geschwindigkeitsvorteil gegenüber der gesamten Konkurrenz macht sich aber so richtig erst dann bemerkbar, wenn es an den Ausdruck in der exzellenten »Near Letter Quality«-Schrift geht. Während die meisten Drucker die NLQ-Schrift nur im unidirektionalen Druck erzeugen können, druckt der DX 2100 weiterhin bidirektional. Das heißt, wenn der Druckkopf über eine Zeile hin und wieder zurückgestrichen ist, ist die Zeile fertig. Das läßt sich naturlich auch mit Zahlen belegen; die Druckgeschwindigkeit beträgt in der NLQ-Schrift immerhin noch 44 Zeichen pro Sekunde und steht damit ebenfalls an der Spitze aller bisher getesteten Drucker.

Neben der exzellenten Schriftqualität haben sich die Konstrukteure des DX 2100 aber noch einige Besonderheiten einfallen lassen. So fällt es beispielsweise sehr angenehm auf, daß man das Endlospapier (einfach durch Druck auf eine Funktionstaste nach hinten transportieren), eingespannt lassen kann, wenn man mit Einzelblättern arbeitet. Sowohl das Endlospapier als auch die Einzelblätter werden automatisch per Tastendruck eingezogen und genauestens justiert. Der Papierantrieb ist dabei so geschickt hinter der Schreibwalze angebracht, daß das Papier auch tatsächlich an der als Abrißkante bezeichneten Stelle sauber abgetrennt werden kann.

Was wäre ein Drucker ohne seine Steuerbefehle? Beim DX 2100 hat man wohl eine der bemerkenswertesten

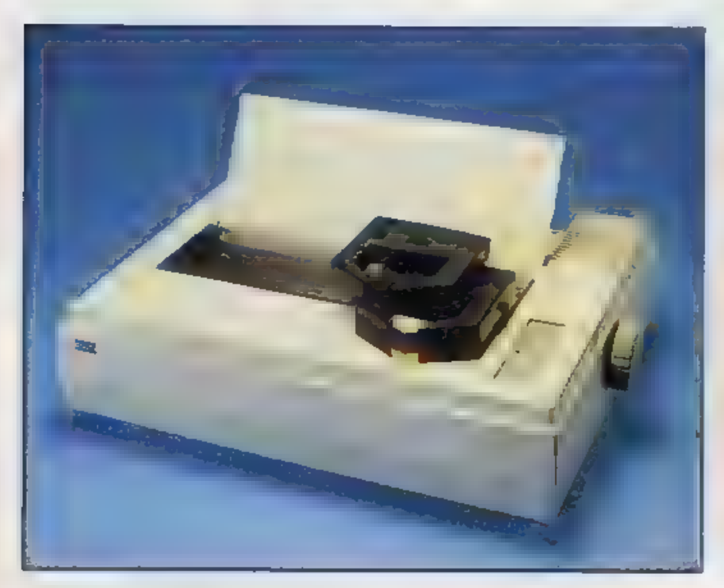


Bild 5. Der DX 2100, Farbe oder Schwarzweiß nach Wahl

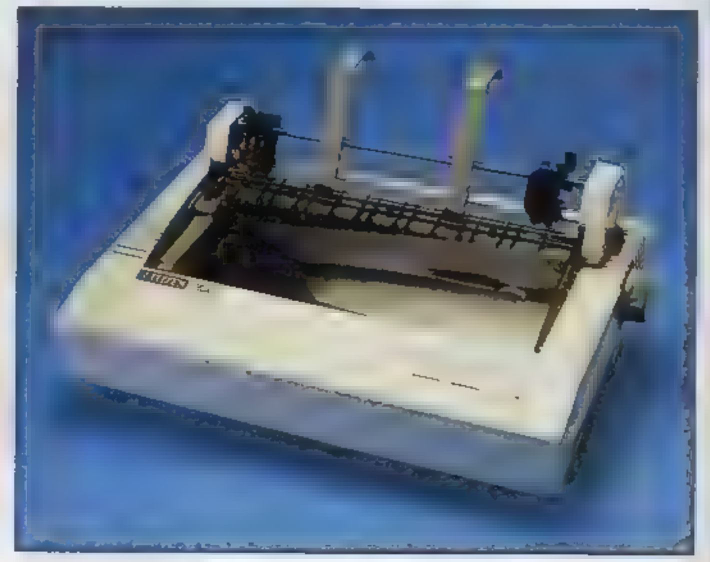


Bild 6. Der Citizen 120 D – ein Kleiner ganz groß

Auswahlpaletten an ESC-Befehlen, denn sowohl die FX-85- als auch die JX-80-Farbsteuerbefehle sind vorhanden (Tabelle). Wer aber lieber die Schriften mit Schaltern oder Tasten einstellen möchte, braucht beim DX 2100 nicht lange zu suchen. Die vier Funktionstasten dienen, je nachdem, in welcher Reihenfolge beziehungsweise Kombination sie gedrückt werden, dazu, 14 verschiedene Schriften beziehungsweise Funktionen aufzurufen oder zu löschen. Alle diese Funktionen sind auf die gängigsten Programme abgestimmt. Dank einem in dieser Form noch nie dagewesenen Steckkartenkonzept kann praktisch jede beliebige Schnittstelle entwickelt werden und in Form einer Karte eingeschoben werden. Auch die Aufrüstung des internen Speichers von 2 KByte ist um 8 KByte oder 16 KByte mit zwei Handgriffen durch einfaches Einstecken der Karte möglich. Da die sonst so gerne versteckten DIL-Schalter beim DX 2100 auf der Speicherkarte angebracht sind, kann man sie zwar nicht unmittelbar, aber immerhin ohne große Umstände und ohne Werkzeug erreichen. Der Zugriff auf die DIL-Schalter ist aber wegen dem weiter oben beschriebenen Bedienereinstellmodus nur in sehr seltenen Fällen notwendig. Einen Nachteil des DX 2100 konnten wir während des Tests allerdings doch noch entdecken: Die Anzeige-LEDs für ON Line, NLQ und Power On sind etwas zu schwach und zu klein.

Mit einem Listenpreis von 1932 Mark und einem Aufpreis von 456 Mark für die Farboption ist der DX 2100 ein Drucker der, gemessen an seinen Leistungen, preiswert ist.

Citizen – der Riesenzwerg

Der 120 D ist ein kleiner Drucker, der einer der ganz Großen werden kann (Bild 6). Nicht ohne Grund, denn in dem kleinen Kerl steckt so manche Überraschung. Gleich nach dem Auspacken des 120 D fällt das sehr kompakte Äußere auf. Mit 37,5 Zentimeter Breite und nur 23 Zentimeter Tiefe ein relativ kleiner Drucker mit geringem Gewicht. Die schon fast üblichen Tasten ONLINE, FF und LF sind mit den optischen Kontrollen auch hier vorhanden. Auf die Funktionen dieser Tasten soll später eingegangen werden. Das Einlegen der Farbbandkassette verursacht zwar kein Kopfzerbrechen, dafür aber ganz schön schwarze Finger. Der Druckkopf macht einen etwas zierlichen Eindruck, ist aber seinen Aufgaben problemlos gewachsen. Von einem Zahnriemen getrieben, wird er auf zwei Lagern geführt. Davon ist das vordere eine Schiene und das hintere ein Winkel als Gegenlager. Damit wird die für die Schönschrift erforderliche Präzision erreicht, obwohl zwei Schienen die bessere Lösung wären. Als ein Drucker mit Epson- und IBM-Modus wird der 120 D vom Handbuch vorgestellt. Das Handbuch ist gut gegliedert und gibt, klar in der Darstellung, auf fast alle Fragen Antwort. Ein deutsches Handbuch soll, nach Auskunft des Importeurs, bis zur Auslieferung der ersten Geräte vorliegen.

Auch der 120 D hat, wie die Geräte der MSP-Serie, einen sogenannten »Maintenance Self Test«, kurz MST, der über die eingebaute ROM-Version und das Datum der Programmierung Auskunft gibt. Anschließend werden Zeile für Zeile »H«s gedruckt, die der Überprüfung der Lage des Druckkopfes dienen. Leider wird die Stellung der DIP-Schalter im MSP des 120 D nicht mehr angezeigt. Dies war angenehm bei dem MSP 10, ist aber beim 120 D auch nicht mehr unbedingt notwendig, da die DIP-Schalter von außen zugänglich sind. Den Entwicklern des 120 D ist zur Frage der Schnittstellen etwas Besonderes eingefallen. Das jeweils gewünschte Interface wird als Kassette eingeschoben. Diese Idee hat man perfektioniert, und so lassen sich Schnittstellen bequem und ohne Werkzeug an der rechten Seite des 120 D einschieben. Die DIP-Schalter befinden sich in dieser Kassette und sind im eingeschobenen Zustand unter dem Druckkopfweg zugänglich. Im Grundzustand ist der 120 D, ähnlich einer Schreibmaschine, nur für Einzelblätter geeignet. Diese lassen sich einfach und sicher einlegen. Wird die Verarbeitung von Endlospapier notwendig, so kann ein zum Lieferumfang gehörender Traktor mit einem Handgriff aufgesetzt werden.

Die Verriegelung der Stachelwalzen ist, gegenüber dem MSP 10, offensichtlich mit gutem Ergebnis überarbeitet worden und ist kein Grund zur Kritik mehr. Wegen des Aufsetzens und dem damit verbundenen längeren Transportweg des Papiers geht allerdings beim Entnehmen eines Dokuments ein Leerblatt verloren.

Eine gute Sache, die beim 120D nochmals verbessert wurde, ist der vom MSP 10 bekannte Hex-Dump. Durch Drücken von LF und FF während des Einschaltens angewählt, wird man beim Ausdruck einer Textzeile überrascht seinen Augen nicht trauen. Nicht nur die Hex-Werte erscheinen auf dem Papier, sondern auf der rechten Seite des Blattes auch die dazu gehörenden ASCII-Zeichen. Ein Leistungsmerkmal, das den MSP10 schon als anwenderfreundlich ausgewiesen hat.

Beim 120 D ist dieser Modus nochmals verbessert. Außer der Darstellung der ASCII-Zeichen werden auch »CR, SPACE, ESC« und andere Steuer-Befehle als Abkürzung dargestellt Beim MSP 10 befand sich beispielsweise an der Stelle des »CR« ein Punkt. Beim 120 D erscheint das Kürzel »CR« so, daß beide Buchstaben übereinanderstehen. Jeder, der schon einmal mit Handbüchern Hex-Werte in ASCII-Zeichen umgesetzt hat, wird diesen Komfort des 120 D schnell schätzen lernen.

Mit den Tasten ONLINE, LF und FF werden Hex-Dump, der Selbsttest, der Maintenance-Test und die Umschaltung in den NLQ-Modus vorgenommen. Das Einschalten des NLQ-Modus kann zeilenweise geschehen, das heißt, wird während eines Druckvorganges in den NLQ-Modus geschaltet, so wird die laufende Zeile ordnungsgemäß beendet, und dann werden die folgenden Zeilen in Schönschrift aufs Papier gebracht Ähnlich dem Epson FX-85 kann mit ONLINE und FF in einen Auswahlmodus geschaltet werden, der dann die Wahl verschiedener Darstellungsarten ermöglicht.

Papierbreiten von 76,2 mm bis hin zu 240 mm sind keine besondere Aufgabe für den 120 D. Dabei sind außer dem Original noch zwei Durchschläge möglich. Die Anschlagstärke wird an einem kleinen Hebel, seitlich rechts am Gehäuse, verändert. Keine Frage, auch Einzelblätter verarbeitet der 120 D mühelos.

Die hinlänglich bekannten, weil oft verwendeten ESC-Befehle dienen dem 120 D wie auch dem Vorbild zur Steuerung von Format-Anweisungen, Schriftartenumschaltung und der Ausführung sonstiger Steuerbefehle. Der 120 D verfügt im wesentlichen über den Befehlsumfang des FX-85, weshalb auch mit den wenigsten Programmen Probleme zu erwarten sind. Ein Befehl soll aber an dieser Stelle besonders hervorgehoben werden: CHR$(27) »h«. Mit diesem Befehl wird der 120 D veranlaßt, alle nachfolgenden Zeichen in doppelter Höhe zu Papier zu bringen.

Fur 998 Mark wechselt der 120D seinen Besitzer und zeichnet sich damit durch ein exzellentes Preis-/Leistungsverhältnis aus. Bedenkt man, daß sich der Marktpreis in der Regel noch etwas unterhalb dieses Listenpreises einpegelt, kann man von einer kleinen Sensation sprechen. Man erhält neben einem sehr guten Schriftbild in Near-Letter-Qualität auch fast alle Funktionen des Epson FX-85 und sogar noch einen erweiterten Hex-Dump dazu. Alles in allem kann der Citizen 120 D ein Verkaufsschlager werden, denn ähnliche Leistungen wurden bislang nicht für so wenig Geld angeboten. (aw)

Scharfer Blick für Könner

Eine neue Woge schwappt über das Land: Ein Heer von Grafik-Freaks brachte uns eine CAD- und Grafik-Software-Schwemme für den semiprofessionellen und den Amateurbereich, daß uns (Radio-) Hören und (Fern-)Sehen verging. Aber für den neuen Genuß brauchen wir einen »scharfen Blick«: Monitore!

Fantastische Zahlen werden von den zukünftigen Computern der 68000er-Familie als Gerüchte laut. Da ist von bis zu 4096 verschiedenen Farben und technischen Daten die Rede, die oft in keinem Verhältnis zu dem bisher Bekannten stehen.

Die Praxis sieht meist erheblich nüchterner aus, als Prospekte vermitteln wollen

Man sollte sich vergegenwärtigen, daß die meisten dieser Computer bis heute lediglich 3-Bit-D/A-Wandler am Ausgang der Videoverarbeitung besitzen. Das gilt also zum Beispiel für analoge RGB-Ausgänge bei einigen dieser Geräte. Aber auch die digitale Codierung wird meist nur mit 3-Bit-Codierern vollzogen.

Welche Bedeutung hat das für die Praxis? Ganz einfach: Die Zahl der möglichen Kombinationen nimmt drastisch ab. Da wir ja alle unsere Hausaufgaben in dualer Rechenweise gemacht haben, wissen wir, daß bei einer Datenbreite

Mit Tricks zu mehr Schärfe

von 3 Bit die maximale Zahl der gleichzeitig darstellbaren Farben $2^3 = 8$ als Multiplikator, also 8 x 8 x 8 = 512 beträgt! Erst bei 4-Bit-Darstellung entspricht das $2^4 = 16$ als Multiplikator, also 16 x 16 x 16 = 4096! Deshalb greift man in den Entwicklerlabors oft zu Tricks. Ein beliebtes Mittel ist das Ersetzen von echter hoher Auflösung durch mehrfache oder erhöhte Darstellungsfrequenz.

Atari arbeitet zum Beispiel mit einer Bildwechselfrequenz von 71 Hz, einem Wert, der über den üblichen gerade noch wahrnehmbaren 50 Hz liegt. Der Vorteil liegt auf der Hand: Man erhält ein gestochen scharfes Bild ohne Flimmern und Flackern. Der Preis der Monitore ist natürlich etwas höher als üblich, da die Video-Bandbreite recht gute Werte haben muß, um dieses Bild darstellen zu können. Wie man sich denken kann, ist es kein leichtes Unterfangen, den Atari an herkömmliche Monitore oder gar Fernsehgeräte anzuschließen. Man sollte mindestens auf eine umschaltbare Synchronisationsfrequenz 50/60 Hz achten, wie sie beispielsweise Sanyo bei einigen Monitoren einbaut.

Bei verschiedenen Fernsehgeräten mit SCART-Buchse ist ein Anschluß aber ohne Probleme möglich, zumindest bei 60 Hz, was auch eigenen Meßergebnissen (RGB-Sync-Frequenz 58,8 Hz!) entspricht.

Die von Atari angebotenen Monitore SM 124 (Monochrom, hochauflösend 640 x 400 Punkte) und SC 1224 (RGB, 640 x 400 Punkte) sind von ausgezeichneter Qualität und ihr Geld mehr als wert, zumal die erwähnte Kompatibilität gewährleistet ist.

512 Farbstufen sind darstellbar, davon maximal 16 gleichzeitig bei einer Auflösung von 320 x 200 Punkten oder bei 640 x 200 Punkten 4 Farben (Vollfarben). Commodore setzt mit Amiga eher auf Kompatibilität zur Fernsehnorm. Die hier verwendete Bildwechselfrequenz beträgt »nur« 60 Hz, was zur Folge hat, daß ein geringer Rest von Flimmern wahrnehmbar bleibt. In der Praxis schlägt aber die hohe Auflösung (640 x 400) an Schärfe die 8-Bit-Konkurrenz um Längen. Das Format von 13 Zoll und eine RGB-Ansteuerung des Monitors ermöglichen so eine einwandfreie 80-Zeichen-Darstellung.

Während der Atari im RGB-Modus an dem dafür vorgesehenen Monitor nur eine Auflösung von 320 x 200 Bildpunkten erlaubt, es sei denn monochrom (640 x 400), stellen sich derartige Probleme dem Amiga-Benutzer nicht. Er kann hochauflösende Grafik mit bis zu 16 verschiedenen Farben erzeugen. Monitore, die dies verkraften, sind allerdings nicht gerade billig. Zwar findet man in den meisten Fällen preiswerte Alternativen, aber oft erst nach langer Suche. Selbst dann noch bleibt dahingestellt, ob es nicht unsinnig ist, einen Computer mit erstklassigen Grafikfähigkeiten an einen (wahrscheinlich) relativ minderwertigen Farbbildschirm anzuschließen.

Der letzte Computer mit eigenem Bildschirm in dieser Reihe ist der MacIntosh von Apple. Beim Umgang mit ihm bleiben solche Probleme erspart: Der hochauflösende 9-Zoll-Monitor ist fest in das Gerät integriert; die Darstellung ist ohnehin nur monochrom. Auch hier wird mit einer Bildwechselfrequenz von 60 Hz gearbeitet.

Bei einer Auflösung von 512 x 342 Punkten und einer Taktfrequenz von 7,8 MHz sind mit dem integrierten Desk-Top-System beachtliche grafische Tricks möglich. Ein Grafik-Programm wird angeboten, das eine verblüffende Verwandtschaft zu dem vom Atari bekannten »Neo-Chrome« aufweist (wenn auch ohne »Chrome«). Noch eine Besonderheit ist der optionale Treiber Mac-Plot, der Plotter unterschiedlichster Art ansteuern kann. Selbst Farbausgabe wird möglich, indem jedem Muster des Macintosh eine bestimmte Farbe zugeordnet wird.

Brillant in jeder Beziehung

Das krasse Gegenteil zu dieser fixen Kombination ist der frei erweiterbare Gepard-PC, wobei grundsätzlich zwei verschiedene sogenannte »Pakete«, nämlich das Textpaket und das Grafikpaket zur Auswahl stehen.

Die Textversion verfügt über eine 80-Zeichen-Karte, die Grafikversion hingegen enthält eine GDC-Karte, die Auflösungen von maximal 1000 x 700 Punkten (!) erlaubt. Der NEC-Grafik-Prozessor bringt zwar nur eine Auflösung von 200 x 170 Punkten, dafür aber 256 Farben.

Schön, schön, aber was ist das gegen den Atari oder gegen den Amiga?

Jetzt kommt aber der Clou des Ganzen. Mit dem nötigen Kleingeld kann der ambitionierte Grafik-Designer sein Grundgerät erheblich erweitern. Mit insgesamt acht GDC-Karten und daraus resultierender 128-KByte-Video-RAM ist die volle Auflösung erreichbar. Man stelle sich vor: 1000 x 700 Punkte mit 256 Farben! Deswegen hat der Gepard mit seinem spartanisch abgemagerten Gerüst selbst bei der Mikroskopie-Darstellung oder im Video-Studio seinen Platz gefunden. Der Systemtakt ist extrem schnell (je nach Ausführung bis 16 MHz) und ermöglicht

auch komfortable Bearbeitungszeiten. Bleibt die Frage nach dem geeigneten Monitor. Für den Einsteiger in die 68000-Welt bietet Gepard bis jetzt zwei monochrome Taxan-Monitore an. Der Preis für Farbmonitore in der angestrebten Qualität reicht bis in schier unerschwingliche Bereiche: von zirka 1 200 Mark bis über die 10 000-Mark-Grenze ist alles geboten, je nach Anforderung. Übrigens: Manchmal ist ein kleiner Farbmonitor besser als ein großer. Die Anzahl der Farbtripel (Leuchtpunkte) ist nämlich konstant. Interessant dürfte in diesem Zusammenhang Sonys 10-Zoll-RGB-Monitor sein, dessen exzellente Schärfe ein wahrer Augenschmaus ist.

Er ist auch preislich im untersten Bereich angesiedelt. Verschiedene Wahlmöglichkeiten lassen den Anschluß sowohl analoger als auch digitaler RGB-Signale zu.

Der Außenseiter in unserer Kollektion ist der Stride. Als technisch-wissenschaftlicher Mikrocomputer mit 520 KByte RAM (Minimum) bis 12 MByte konzipiert, verwendet der Stride unter anderem eine Zusatzkarte mit einer Auflösung von 784 x 325 Punkten. Die anderen Steckkarten mussen alle zusätzlich hinzugefügt werden, was dem einen oder anderen langsam klarmacht: Hier geht es um Geld. Die oben erwähnte Grafikkarte arbeitet mit dem extern anzuschließenden Terminal »Wyse 50«.

Der letzte im Bunde ist Sinclairs QL. Die maximale Zahl der Zeichen pro Zeile beträgt 85 statt 80, ein recht ungewöhnliches Format. Die Grafikauflösung beträgt 256 x 256 Punkte (acht Farben) oder 512 x 256 Punkte (vier Farben). Mit knapp 1 700 Mark ist der QL auch der preiswerteste unter den Fünfen. Für den QL wird ein auf ihn abgestimmter Farbmonitor von Sinclair angeboten. Der QL besitzt darüber hinaus aber auch einen eingebauten Modulator für den Ferseh-Kanal 36 (UHF) gleich neben dem ZDF und ist damit an jeden PAL-Fernseher anschließbar. Manche Geräte haben allerdings Probleme mit dem Synchronsignal des QL, das unverständlicherweise invertiert ist. Im Zweifelsfall muß der Fachhändler helfen. Erfahrungsgemäß wird der QL sowieso fast nie als Einsteigermodell gekauft und gelangt daher meist in berufenere Hände. Der Käufer kommt in der Regel aus der »Sinclair-Ecke«, kennt also derartige Dinge schon vom Spektrum, dessen Schwarzweiß-Signal ebenfalls invertiert ist.

Machen wir jetzt einen kurzen Abstecher in die Theorie.

Es gibt grundsätzlich drei völlig verschiedene Prinzipien von Bildschirmen mit unterschiedlicher Zielsetzung.

1. Speicherbildröhren:

In der Bildröhre dieses Typs befinden sich Phosphor-Partikel mit extremer Nachleuchtdauer bis zu einigen Stunden. Grafische Designs bleiben ohne jeden Flimmereffekt über lange Zeit

Zur Theorie:

erhalten. Dabei ist von Nachteil, daß für jede geringfügige Änderung der gesamte Bildschirm gelöscht werden muß, was durch einen kurzen Anoden-Spannungs-Impuls ausgelöst wird. Man muß also Genauigkeit vor den Faktor Zeit setzen. Aber es lohnt sich zum Beispiel für Trickfilmer. Durch die extrem feine Bündelung der Elektronenstrahlen ist eine ultrahohe Auflösung von 4 096 x 4 096 Punkten möglich! Monitore mit diesen Röhren sind Profigeräte, sehr teuer und nicht ohne weiteres an Heim-Computer anschließbar.

2. Vektor-Refresh-Displays:

Hier wird die Speicherung in Koordinaten von x,y festgehalten und stets neu geschrieben, bis eine Änderung eintritt. Teile des Bildschirms sind mit mittlerer Geschwindigkeit änderbar. Die Ursache liegt in der benötigten Rechenzeit und im Speicherplatzbedarf. Bei zu schnellen Änderungen entsteht der Eindruck des Wischens wie etwa bei billigen Video-Kameras. Auch hier sind 4 096 x 4 096 Zeichen möglich, aber bei eingeschränkter Farbenzahl und fehlender Flächendarstellung. Dieser Typ ist mit Heim- und Personal Computern unverträglich.

3. Raster-Scan-Monitor:

Der verbreitetste Monitortyp benutzt wie Fernsehgeräte ein Abtastraster, das der Elektronenstrahl stur entlangfährt, im Gegensatz zu den vorangegangenen Displays, deren Elektronenstrahl beliebig positioniert werden kann. Die maximal mögliche Auflösung ist hier deutlich niedriger: nur beim Monochrom-Schirm sind 2048 x 2048 Punkte darstellbar. Das Bild wird immer wieder neu »geschrieben«. Beim Zeilensprungverfahren (Interlace) werden pro Sekunde 50 Halbbilder erzeugt, beim Non-Interlace-Verfahren werden 50 Vollbilder pro Sekunde erzeugt. Bei der Übertragung und beim Aufbau des Bildes wird ein ziemlicher Aufwand notwendig, der eine hohe Auflösung aus Gründen zu hoher Fertigungskosten verhindert. Der unbezahlbare Vorteil ist jedoch, daß jedem Punkt eine Intensitätsinformation mitgegeben werden kann. Durch die Darstellungsmöglichkeit von bis zu 256 Helligkeitsgraden können auch mit relativ wenigen Farben aufregende Bilder mit 3D-Effekt hergestellt werden. Außerdem ist es nur hier möglich, Ebenen zu überlagern, wie etwa Transparentgrafiken.

Damit die Farbstrahlen nicht »fremdgehen«, werden sie durch Lochmasken oder Schlitzmasken geschickt. Bei reinen Lochmaskenmonitoren sind die Strahlen oft nicht deckungsgleich. Es entstehen sogenannte Konvergenzfehler, im Volksmund unter dem Spitznamen »Japaneffekt« bekannt, in Erinnerung an die Plastik-Ferngläser der Kindheit.

Die Schlitzmaske allein ist auch keine Lösung, da sie ein größeres Raster und somit eine reduzierte Auflösung bedingt. Die als PIL-Technik (Precision-In-Line) eingeführte, neuere Methode der Kombination aus Schlitzmaske und Farbtripeln in der altbewährten 120-Grad-Anordnung, der sogenannten Delta-Beschichtung, gibt berechtigten Anlaß, von einer neuen gelungenen Monitor-Generation zu sprechen.

Man kann im Mittel von einer Auflösung von etwa 1 280 x 1 024 Bildpunkten ausgehen. Höhere Qualität kann nur durch absolut überteuerte Fertigungskosten erreicht werden, die niemand bezahlen würde. Diesem Monitortyp entsprechen praktisch alle Monitore für Heim- und Personal Computer.

Acht Farben, das reicht

Zurück zu unserem Ziel, der Grafik-Darstellung am 16-Bit-Computer. Wir wissen bereits, daß eine Beschreibung mit 3 Bits (pro Pixel) acht reine Vollfarben (Schwarz, Rot, Grün, Blau als Primärfarben sowie die einfachen Kombinationen der additiven Lichtfarbenmischung Gelb, Magenta, Cyan und Weiß) darzustellen ermöglicht sowie deren Intensität.

Ob man's glaubt oder nicht, für 90 Prozent aller Computer-Grafiken reicht das aus. Es ist sogar faszinierend, wieviele Farbschattierungen sich dadurch erzielen lassen.

Im Kampf um Marktanteile wird sich der Trend zur Individualisierung bei den Videodaten der verschiedenen Computertypen immer stärker durchsetzen. Neue Geräte und neue Formate unabhängig von einer starren, durchaus notwendigen Fernsehnorm schaffen bessere Bildqualität und Markenbindung in einem. Vielleicht wird ja auch eines Tages unsere betagte Fernsehnorm erneuert und kann von dieser Entwicklung profitieren. Harren wir also der Dinge, die da kommen, aber warten wir nicht zu lange. Schließlich ist auch das modernste Gerät irgendwann antiquiert, und in 20 Jahren kräht wahrscheinlich kein Hahn mehr nach einem dieser genialen Geräte.

(Helmut Jungkunz/lg)

Kauderwelsch

Für die 68000er hat sich noch kein Betriebssystem etabliert. Bisher brodelten bei den Software-Alchimisten überall unterschiedliche Süppchen.

Als Sir Clive Sinclair den »QL« der Presse präsentierte, war eine der ersten Fragen: Warum hat Sinclair für den QL mit QDOS ein neues Betriebssystem entwickelt? Die Antwort lautete damals: Weil es kein Betriebssystem gibt, das die Fähigkeiten des QL voll unterstützt.

Viele Hersteller denken ähnlich. Das ist einer der Gründe, warum keiner der neuen Computer mit »68000-Herz« über ein Betriebssytem verfügt, das sich bereits in dieser Computergeneration als Standard etablieren konnte. Das liegt unter anderem an spezifischen Bausteinen, die jeder Hersteller für seinen Computer entwickelte. Das Betriebssystem soll natürlich besonders effizient sein und geht daher in besonderer Form auf die Eigenarten der Hardware ein. Manchmal ist auch die Softwareumgebung Ursache für diese Eigenbrötlerei. So war zum Beispiel der Macintosh von Apple einer der ersten Computer mit einer Benutzeroberfläche – und das Betriebssystem wurde auf diesen »Komplizen« zugeschnitten. Benutzeroberflächen haben natürlich ihre Berechtigung und sind förderungswürdig; daß sich aber das Betriebssystem an diese darübergelegte »Schale« anpassen muß, ist wohl übertrieben. Das hat Digital Research eindrucksvoll bewiesen. »GEM« nämlich arbeitet mit einer Anzahl von unterschiedlichen Betriebssystemen zusammen.

Newcomer

Das »TOS« des Atari ST entwickelte Digital Research. Es ist ein Ableger von CP/M-68. Leider entbehrt es dessen Multitasking-Fähigkeit, obwohl der 68000-Mikroprozessor, den Atari bei der ST-Serie verwendet, für Multitasking geradezu prädestiniert ist.

Aber »TOS« hat andere Vorteile zu bieten. So ist zum Beispiel die hierarchische Dateistruktur leistungsstark und gestaltet die einfache und übersichtliche Verwaltung von vielen Dateien durch Aufteilung in Unterverzeichnisse. Besonders bei der Arbeit mit einer Festplatte, die Atari noch in diesem Jahr auf den Markt zu bringen beabsichtigt, ist das von Vorteil Auf Einzelheiten kommen wir noch zu sprechen.

Der Amiga, Commodores neues Wunderkind, wartet auch mit einem neuen Betriebssystem auf. Laut Dr. Tim King, dem Software-Chef von Metacomco und Entwickler von Amiga-DOS, soll die Leistungsfähigkeit seiner Dateiverwaltung zwischen MS-DOS und Unix liegen. Integrierte Grafikroutinen verwandeln den Amiga in ein begeisterndes Werkzeug für Grafik-Enthusiasten. Eingeweihte Kreise sehen darin den Anwärter eines neuen Betriebssystems in der 68000er-Klasse. Multitasking und die leichte Übertragbarkeit des in »C« geschriebenen Systems sprechen dafür.

Das QDOS von Sinclairs Flaggschiff hat durch die direkte Verbindung mit Basic und einer unglücklichen Geschäftspolitik eine eventuelle Marktposition verspielt. Als der QL in England angekündigt wurde, sprachen die inländischen Softwarehäuser von einem neuen Stern am Computerhimmel. Einer der Hauptgründe war das einfach zu programmierende Multitasking des QL und die starke Position von Sinclair Research Ltd Dieser britische Hersteller hielt fast 70 Prozent des englischen Heimcomputer-Marktes in Händen. Nachdem sich Experten das Gerät näher betrachtet hatten, zerplatzte der Traum des neuen Sterns wie eine Seifenblase.

Schuld daran war die mangelhafte Hardware. Kleinen Bandlaufwerken und einer äußerst gewöhnungsbedürftigen Tastatur traute man keine großen Erfolge zu. Massive Probleme, versteckte Fehler in QDOS auszumerzen, taten das übrige. Ob der Nachfolger des QL, von dem Insider bereits munkeln, Chancen hat, sich zu behaupten, kann jetzt noch niemand wissen.

Apple kreierte mit dem Macintosh einen der ersten Computer mit Benutzeroberfläche. Allerdings ist diese Benutzeroberfläche, mit Namen »Finder«, stark mit dem Betriebssystem verschmolzen. Geheimniskrämerei um diese sogenannte »Black Box« und eine nach innen gerichtete Firmenpolitik verhinderten eine Implementation auf andere Computer.

So warten die Computerhersteller auf einen Vorreiter – einen Vorreiter, der sein Betriebssystem etabliert und damit ein großes Softwarepotential an Anwendersoftware nach sich zieht, auf das andere aufspringen können. Eindrucksvoll hat das IBM mit ihrem Personal Computer bewiesen. Daß dieser Computergigant als »Zugpferd« fungiert, kann man leider gänzlich ausschließen. Seine Produktpolitik weist unveränderlich in Richtung Intel und deren Mikroprozessorreihe. Auch das große Softwarehaus Digital Research hat sein Multitasking-System CP/M-68 nicht in der Computerindustrie durchgesetzt. Zwar haben andere Firmen Anleihe genommen – zum Beispiel basiert »TOS« auf CP/M-68 –, kompatibel sind beide Betriebssysteme trotzdem nicht, und das ist der große Haken. Was hilft es denn, daß eine gewisse Anzahl von Aufrufen für Systemroutinen gleich sind. Software, die für CP/M-68 entwickelt wurde, gewinnt trotzdem nur durch einen Emulator ihre Lauffähigkeit. Dieselbe Ablaufgeschwindigkeit wie unter dem Betriebssystem ist dadurch nie zu erzielen

Welches Betriebssystem das Rennen machen wird, steht also noch in den Sternen. Leider kommen immer neue Anwärter auf den Markt, und von Kompatibilität kann keine Rede sein. Wie in vielen Branchen ist der Endkunde der »Gelackmeierte«. Die Software, die er für einen Computer mit 68000er-Prozessor heute kauft, kann für den Nachfolge-Computer unbrauchbar sein. Es bleibt zu hoffen, daß immer mehr Hersteller ihren Computern diese leistungsstarke »Central Processing Unit« einverleiben und sich ein Betriebssystem durchsetzt. Noch ist es aber nicht so weit. Deshalb gehen wir im folgenden auf die vorhandenen Betriebssysteme etwas näher ein.

»TOS« – Tolles Operating System?

TOS ist die Abkürzung für »Tramiel Operating System«. Bei dieser Bezeichnung dachten viele: Schon wieder ein neues Betriebssystem! Natürlich ist das richtig, aber es weist auch viele Gemeinsamkeiten mit bereits etablierten Systemen auf.

In nur fünf Monaten stellte das Entwicklungsteam mit dem Atari ST einen revolutionären Computer auf die Beine, der es in sich hat. In dieser kurzen Zeit ein völlig neues Betriebssystem zu entwickeln, ist kaum durchführbar. Man wandte sich an einen großen Softwarehersteller, der bereits etwas Fertiges in der Schublade hatte. Dieser Softwarehersteller hieß Digital Research. CP/M-68 brauchte »nur« angepaßt zu werden, einige Leistungsmerkmale blieben dabei leider auf der Strecke. Eines davon ist, wie schon erwähnt das Multitasking. Obwohl der ST mit dem 68000-Mikroprozessor eine CPU aufweist, die Multitasking quasi »mit links« macht, wurde es nicht implementiert. In vielen anderen Dingen gleicht TOS aber sehr dem CP/M-68.

So sind zum Beispiel viele Einsprünge für Routinen identisch – viele, aber nicht alle. Kompatibel sind TOS und CP/M-68 leider nicht.

TOS besteht aus mehreren Teilen: dem Kern, dem BIOS und dem XBIOS. Der Kern enthält alle Routinen zur Verwaltung des Dateisystems. Das Dateisystem ist komfortabel und durch die hierarchische Struktur auch in der Lage, eine große Anzahl von Dateien zu verwalten.

Die TOS-Systemfunktionen

Nr.	Funktion
0	Terminiere Programmausführung
1	Konsol-Zeichen lesen (mit Echo)
2	Konsol-Zeichen schreiben
3	AUX-Kanal-Zeichen lesen
4	AUX-Kanal-Zeichen schreiben
5	Drucker-Zeichen schreiben
6	Direkte Konsol-Zeichen lesen/schreiben (ohne Echo)
7	Direkte Konsol-Zeichen lesen (ohne Echo)
8	Konsol-Zeichen lesen (ohne Echo)
9	Konsol-String schreiben (nullterminiert)
10	Konsol-String lesen
11	Konsol-Eingabe Status
13	Laufwerk zurücksetzen
14	Default-Laufwerk setzen
25	Default-Laufwerk ermitteln
26	Disk-Transfer-Adresse setzen
42	Datum lesen
43	Datum setzen
44	Uhrzeit lesen
45	Uhrzeit setzen
47	Disk-Transfer-Adresse ermitteln
48	Versionsnummer ermitteln
49	Prozeß abbrechen und resident halten
54	Freien Speicherplatz eines Laufwerks ermitteln
57	Subdirectory anlegen
58	Subdirectory löschen
59	Directory-Pfad definieren
60	Datei erzeugen
61	Datei eröffnen
62	Datei schließen
63	Datei lesen
64	Datei schreiben
65	Datei löschen
66	Datei positionieren
67	Dateiattribute lesen/schreiben
69	Duplikat von Datei-Handle erzeugen
70	Datei-Handle zuweisen
71	Directory-Pfad ermitteln
72	Speicherbereich dynamisch anfordern
73	Angeforderten Speicherbereich zurückgeben
74	Speicherbereich freigeben
75	Programm laden/ausführen
76	Prozeß terminieren
77	Rückkehrstatus eines Prozesses ermitteln
78	Ersten Directory-Eintrag suchen
79	Nächsten Directory-Eintrag suchen
86	Datei umbenennen
87	Zeit/Datum-Attribut einer Datei lesen/schreiben

Die TOS-Systemroutinen mit den Funktionsnummern

Die hierarchische Dateistruktur erlaubt es, Unterverzeichnisse anzulegen. Besonders bei der Arbeit mit einer Festplatte lernt man das sehr zu schätzen. Wir haben die Tiefe der zu definierenden Unterverzeichnisse bis 20 getestet, also in dem Verzeichnis ein Unterverzeichnis angelegt, in diesem ein nächstes und so weiter. 20 erschienen uns als ausreichend. Leider kann man nicht, wie in MS-DOS-Versionen ab 2.11, von einem Unterverzeichnis in ein anderes springen. Man ist gezwungen, jedesmal ins Hauptverzeichnis zurückzugehen, und kann dann erst in das gewünschte Verzeichnis gelangen. Um eine Datei anzusprechen, braucht man sich nicht in dem gewünschten Unterverzeichnis zu befinden. Durch die Angabe des Verzeichnisses oder eines Pfades läßt sich eine Datei auch direkt verwenden. Das gestattet allerdings nur der Editor von TOS. Die Benutzeroberfläche GEM bietet das nicht, da sie Dateien in Unterverzeichnissen nicht anzeigt.

Der TOS-Editor ist sehr einfach und zweckmäßig gestaltet. Eingaben kann man nur durch Löschen von Zeichen korrigieren, genau wie man es von MS-DOS kennt. Dateien lassen sich am Bildschirm anzeigen oder auch ausdrucken. Die Bildschirmausgabe steuern Control-Codes. Mit Control-S hält man das Auflisten an, mit Control-Q läuft es weiter, und durch Control-C bricht es ab. Bei der Arbeit mit Dateien zeigt TOS die gewohnten Leistungen. Jede Datei läßt sich mit einem 8stelligen Namen und einer 3stelligen Kennung versehen, die auch aus Zahlen bestehen darf. TOS prüft beim Speichern, ob die Datei bereits vorhanden ist. Wenn ja, vergewissert es sich durch eine Rückfrage, ob man die vorhandene Datei wirklich überschreiben möchte. Der TOS-Kern verwaltet auch alle Ein- und Ausgabekanäle. Dazu gehören die Disketten- und Festplatten-Ports. Auch die Speicherverwaltung obliegt dem Kern. Durch die Benutzeroberfläche, die ihrerseits aus verschiedenen Teilen besteht, und Anwenderprogramme kann sich manchmal eine größere Anzahl von Programmen im Speicher befinden. Nicht nur die Speicheraufteilung, sondern auch die Verwaltung der Programmabläufe untersteht der Aufsicht dieses Systemteiles. Das betrifft auch die Adresse des Bildschirmspeichers, die nicht festgelegt ist, sondern sich je nach gewählter Größe verschiebt.

Der TOS-Kern benutzt die Routinen des TOS-BIOS. Darin sind alle Standard-Routinen zusammengefaßt, die die Hardware betreffen, wie: Disketten-, Festplattenlaufwerke, Tastatur, Maus, Centronics-, RS232-Schnittstelle sowie die Ansteuerung für Monochrom- und Farbmonitor.

Spezielle Routinen, über die CP/M-68 normalerweise nicht verfügt, enthält der kleine XBIOS-Teil. Eine davon steuert zum Beispiel die MIDI-Schnittstelle.

Hallo, System

Um TOS-Routinen aufzurufen, werden vorher die Parameter und die Funktionsnummer auf dem Stack abgelegt. Der Aufruf erfolgt über Trap #1. Folgt dem Aufruf eine Fehlermeldung, steht diese in Register D0. Angezeigt wird eine Fehlermeldung durch den Wert FFFF im höherwertigen Wort des Registers D0. Im niederwertigen steht die Fehlernummer. Vor der Verwendung muß man sie negieren.

Um BIOS-Routinen aufzurufen, benutzt man Trap #13, was aber nur Erfolg hat, wenn sich der Prozessor im Supervisor-Mode befindet. Zuerst übergibt man die Funktionsnummer, dann die Parameter und den Device-Code. Der Funktionsaufruf der XBIOS-Routinen entspricht im Aufbau den BIOS-Aufrufen. Dazu dient Trap #14. Auch hier muß sich der Prozessor im Supervisor-Mode befinden, da sonst ein Busfehler auftritt.

»C«-Telefon

Das Betriebssystem spricht sowohl auf Assembler als auch auf einen C-Compiler an. Da das Betriebssystem auch in C geschrieben wurde, ist die Parameterübergabe dieser Programmiersprache angeglichen.

Befehl	Funktion
TYPE	Datei anzeigen
ERASE	Datei löschen
DIR	Inhaltsverzeichnis anzeigen
RENAME	Datei umbenennen
DELETE	Datei löschen
COPY	Datei kopieren
MAKEDIR	Unterverzeichnis anlegen
CHANGEDIR	Unterverzeichnis wechseln
REMAKEDIR	Unterverzeichnis löschen
SHOW	Systemressourcen anzeigen
INIT	System initialisieren
EXIT	Rückkehr zu GEM

Befehle des TOS-Betriebssystems

Etwas anders liegen die Dinge beim QL. Der QL arbeitet mit dem 68008 als Hauptprozessor und einem Zweitprozessor vom Typ 8049.

Der 68008 verfügt über den vollen Befehlssatz des 68000, hat jedoch nur einen 8 Bit breiten Datenbus gegenüber der 16-Bit-Busbreite des

68000ers. Dadurch erhöht sich die Ausführungszeit der Befehle geringfügig (zirka 7 Prozent), gleichzeitig verringert sich der Hardwareaufwand.

Das QDOS-Betriebssystem

Der QL verfügt über zwei eingebaute Microdrive-Laufwerke. Speichermedien sind hierbei Kassetten, in denen ein Magnetband in einer Endlosschleife läuft. Sinclair sah sich somit vor die Aufgabe gestellt, ein eigenes Betriebssystem zu entwickeln, das all diesen Besonderheiten Rechnung trug: QDOS.

Da sich das Betriebssystem mit dem sogenannten SuperBasic-Interpreter den Platz im ROM teilt, ist es oftmals schwer, eine klare Trennlinie zwischen dem Betriebssytem und den Basic-Routinen zu ziehen. Andererseits ist es gerade hierdurch möglich, die Routinen des Betriebssytems einfach vom Basic aus anzusprechen.

QDOS verwendet zum Aufruf der Routinen sowohl die TRAP-Befehle als auch Routinen, die es über Vektoren aktiviert. TRAP-Kommandos, die automatisch in den Supervisor-Modus umschalten, eignen sich besonders für häufig benötigte und zeitkritische Funktionen, da in diesem Betriebszustand die Interrupts abgeschaltet werden.

Das Betriebssystem QDOS wurde für Multitasking ausgelegt. Jedes Einzelprogramm besitzt eine Jobkennung und eine Priorität, die bestimmt, wieviel Rechenzeit ihm zusteht. SuperBasic beispielsweise hat die Job-Kennziffer 0 mit der Priorität 32. Obwohl die Struktur des M68008, insbesondere die programmzählerrelative Adressierung, dem Multitasking besonders entgegenkommt, ist es nicht verwunderlich, daß ein großer Teil des Betriebssytems der Verwaltung der einzelnen Jobs dient. Hier ist besonders der Scheduler zu erwähnen, der die Zeitaufteilung unter Berücksichtigung der Jobprioritäten vornimmt, sowie die QDOS-Routinen, die für die Speicherzuteilung und die Jobverwaltung zuständig sind.

Der Scheduler wird durch einen Hardware-Interrupt jede 1/50 Sekunde oder auch bei der Installation eines neuen Jobs aufgerufen. Jeder Job erhält eine zeitweilige Priorität. Bei jedem Scheduler-Aufruf erhöht sich diese um den Wert der Jobpriorität. Nun wird diese zeitweilige Priorität vom Scheduler mit den Prioritäten aller vorhandenen Jobs verglichen, und der Job mit der höchsten augenblicklichen Priorität kann bis zum folgenden Scheduleraufruf über den Prozessor verfügen. Der laufende Job erhält dann vorübergehend die Priorität 0, und der Zyklus beginnt von neuem. Die QDOS-Routinen zur Job-Verwaltung ruft man mit dem Trap #1-Befehl auf.

CALL <Adresse>,<Parameter>	Ruft ein Maschinenprogramm auf und übergibt die Parameter an die Register.
EXEC MDV<1-8>,<Dateiname>	Lädt Maschinenprogramm und ruft es als Job auf.
EXEC_W MDV<1-8>,<Dateiname>	Lädt Maschinenprogramm und ruft es als Job mit höchster Priorität auf.

Die Systemaufrufe des Sinclair QL

Mit Trap #2 teilt das Betriebssystem Ein-/Ausgabekanäle zu. Der QL verwendet Kanäle zum Datentransport von oder zu den Systemeinheiten wie Bildschirm, Tastatur oder Microdrives. Mit Befehlen, die über Trap #3 aufgerufen werden, sind nützliche Routinen verbunden, die die zuvor definierten Kanäle verwenden. »Open #5,scr_« bewirkt beispielsweise eine Zuordnung des Kanals mit der Nummer 5 zum Bildschirm (Screen). Soll nun ein Fenster (Window) geschaffen werden, reicht die Angabe des Kanals zur eindeutigen Zuordnung aus: »WINDOW #5,50,50, 30,30«. Dadurch öffnet sich ein Fenster mit einer Größe von 50 x 50 Bildpunkten an der Position 30,30. Ausgaben in diesem Fenster erfolgen mit der Anweisung »PRINT #5«. Dieses Beispiel verdeutlicht bereits die enge Verknüpfung zwischen SuperBasic und QDOS.

Trap #4-Befehle verwendet das System für den Basic-Interpreter. Die weiteren Trap-Anweisungen beansprucht QDOS nicht. Sie stehen dem Anwender zur Verfügung. Durch Basic-Befehle lassen sich diese Traps sehr einfach ansprechen. Möchten Sie eine Maschinencode-Routine als Task starten, dann geben Sie »EXEC« und den Namen der Routine an. QDOS startet die Routine als einen Multitasking-Job. Wo freier Speicherplatz zur Verfügung steht, weiß das System. Benötigt der Job während seines Laufs mehr Speicher, als verfügbar ist, stoppt ihn das QDOS.

Vektorisierte Routinen erfüllen eine Vielzahl verschiedener Aufgaben, beispielsweise arithmetische Hilfsprogramme.

Durch die Zuordnung der QDOS-Routinen zu bestimmten Trap-Befehlen wird gewährleistet, daß verschiedene QDOS-Versionen keine Softwarekompatibilitäts-Probleme aufwerfen. Dadurch, daß das QL-SuperBasic als Job behandelt wird, kann man ihm auf einfache Weise neue Befehle anfügen. Besonders Hardware-Entwickler wissen diesen Vorteil zu schätzen. So gibt es bereits zahlreiche Floppy-Disk-Controller für den QL, die mit ähnlicher Syntax wie die Microdrives angesprochen werden. Das Betriebssystem QDOS ist aufgrund der Gegebenheiten des QL natürlich zu keinem der verbreiteten Standards kompatibel.

AMIGA-DOS: freundliches DOS für die »Freundin«

Der Amiga präsentiert sich dem Anwender bekanntlich mit der Benutzeroberfläche »Workbench«. Sie erlaubt es, praktisch alle Kommandos mit der Maus zu aktivieren. Der fortgeschrittene Benutzer und Programmierer wird aber bald das Bedürfnis verspüren, seine Befehle auch über die Tastatur zu erteilen. Für diesen Zweck gibt es das »CLI«-Programm. »CLI« steht für »Command Line Interface«. Bevor man aber mit dem CLI arbeiten kann, muß man dafür sorgen, überhaupt an dieses Programm heranzukommen. Dazu muß im Preferences-Menü die CLI-Option auf ON gestellt werden, da die CLI-Option unter normalen Umständen abgeschaltet ist. Commodore ist nämlich der Meinung, für den unbedarften Anwender reicht die Workbench aus. Wenn der CLI in Preferences aktiviert ist, kann man ihn aufrufen. Dazu ist die System-Schublade auf der Workbench-Disk anzuklicken. Im System-Fenster erscheint nun das Icon für den CLI. Wiederum durch Anklicken dieses Icons taucht das CLI-Fenster auf und das Amiga DOS ist startbereit.

Bevor man loslegt, sollte man sich etwas mit dem Dateisystem des Amiga befassen. Es hat eine baumförmige Struktur. Am Anfang steht das sogenannte »Root-Directory« der Diskette. Dieses Inhaltsverzeichnis kann Dateien und Verzeichnisse von Unterdateien enthalten. Diese Inhaltsverzeichnisse sind genauso aufgebaut wie die Root und lassen sich beliebig schachteln. Um eine Datei zu spezifizieren, muß man nicht nur den Dateinamen angeben, sondern auch den Pfad, den man von der Root aus durch den Baum einschlagen muß, um die Datei zu erreichen. Die Inhaltsverzeichnisse, die man passiert, werden durch Schrägstriche

getrennt angegeben. Wenn man mehrere Disketten-Laufwerke hat, muß man auch das Laufwerk mit angeben. Diese Angabe steht, durch einen Doppelpunkt getrennt, dem Pfad voran. Die Laufwerke spezifiziert man mit DFn, wobei n für die Nummer des Laufwerkes steht. Momentan unterstützt Amiga-DOS die Laufwerke DF0 bis DF3. Ein vollständiger Dateiname wäre zum Beispiel »DF1:HAUPT-DIRECTORY/ENTWICKLUNG/TEST-PROGRAMM«. Hat man mehrere Inhaltsverzeichnisse verwendet, ergeben sich ganz schön lange Dateinamen. Deshalb gibt es das »Current Directory«, das mit dem Befehl »CD« bestimmt und abgefragt werden kann. Wurde ein aktuelles Inhaltsverzeichnis gesetzt, kann man sich den Pfadnamen sparen, Amiga-DOS sucht nach dem File grundsätzlich in der aktuellen Liste. Benötigt man eine Datei aus einem Unterverzeichnis, ist nur der Pfadname ausgehend von diesem Verzeichnis anzugeben. Einen Schritt zurück im Baum bewirkt der Schrägstrich. Was bei MS-DOS oder Unix-Benutzern Verwirrung stiften dürfte, da dort nämlich der Schrägstrich in Verbindung mit dem Rootdirectory steht. Leider kann man in Amiga-DOS keinen allgemeinen Pfadnamen definieren, der immer dann eingesetzt wird, wenn keine spezielle Angabe existiert. Dies ist in MS-DOS mit »PATH« gegeben.

»Devices« und allerlei Seltsames

Neben den Disketten-Laufwerken kennt Amiga-DOS noch einige andere Geräte, die »Devices«. Mit »PTR« spricht man einen Drucker an. Für die Schnittstellen gibt es die Bezeichnungen »PAR« und »SER«. Eine RAM-Floppy definiert man mit RAM und spricht sie auch so an. Die imaginären Geräte »CON« und »RAW« sind selbstdefinierte Fenster, die allerdings mit Einschränkungen zu genießen sind. Das RAW-Fenster gibt nur einen Sinn in Verbindung mit eigenen Programmen. Das CON-Fenster schließt sich, sobald es ein »End of FILE«-Zeichen empfängt. Dadurch ist es zum Beispiel nicht möglich, das Inhaltsverzeichnis in einem Fenster zu listen, da es sofort nach dem Listen wieder vom Bildschirm verschwindet. Auf die Fenster greift man deshalb besser aus Programmen zu. Zuletzt gibt es noch das Gerät »NIL«. Es tut nichts anderes, als alle Werte, die man ihm übergibt, sofort zu vergessen. Will man aus »NIL« lesen, erhält man nur ein »End of File«. Dieses »Device« steht dem Programmierer zu Testzwecken zur Verfügung. Mit dem Befehl »ASSIGN« kann man auch an Directories einen Devicenamen vergeben und diese wie ein Disketten-Laufwerk ansprechen.

Zur Pflege des Dateisystems bietet Amiga-DOS eine Reihe von Befehlen. Der wichtigste ist natürlich »DIR« zur Ausgabe des aktuellen oder eines angegebenen Inhaltsverzeichnisses. Mit »OPT A« erhält man einen Überblick über alle Directories auf dieser Diskette. »OPT I« eröffnet die Möglichkeit, diese Verzeichnisse im Dialog mit dem Amiga durchzusehen. Der Befehl »LIST« erzeugt ein ausführliches Verzeichnis. Mit »INFO« kann man sich einen Überblick über alle Laufwerke verschaffen. Natürlich gibt es das übliche Löschen, Umbenennen, Kopieren von Files. Textdateien anzusehen oder auf den Drucker auszugeben erlaubt das Kommando »Type«.

Neben den Befehlen zum direkten Umgang mit Dateien bietet Amiga-DOS noch einige Hilfsmittel an. Das wichtigste ist der Editor »ED«. Mit diesem bildschirmorientierten Editor kann man Textfiles bearbeiten. »ED« arbeitet mit vielen Befehlen, wie Blockoperationen, die von Textverarbeitungen her bekannt sind. »ED« ist einfach und komfortabel zu bedienen.

Für zeilenweise aufgebaute Dateien gibt es noch einen anderen Editor, den »EDIT«. Auch dieser verfügt über zahlreiche Kommandos wie »SORT« und »SEARCH«, die das Sortieren oder das Suchen in einfachen Dateien erlauben.

Multitasking

Auch Multitasking unterstützt das System. Gibt man vor einem Kommando das Wort »RUN« ein, dann läuft der Befehl im Hintergrund. Mit »RUN TYPE Brief TO PRT:« kann man zum Beispiel einen Text ausdrucken und gleichzeitig einen anderen Text im Editor bearbeiten.

Wenn man Amiga-DOS mit anderen Betriebssystemen vergleicht, so schneidet es nicht besonders günstig ab. MS-DOS zum Beispiel bietet zwar kein Multitasking, übertrifft es aber an Komfort und Flexibilität erheblich. Außerdem bietet Amiga-DOS dem Programmierer eine Unmenge von Routinen an, die er über maschinennahe Programmierung ansprechen kann. Die Fähigkeiten dieses neuen Betriebssystem schlummern also etwas im verborgenen. (Gerhardt Pehland/hb)

Befehl	Funktion
ASSIGN	Zuweisung eines logischen Gerätenamens an ein Directory
BREAK	Setzen des Unterbrechungsflags für eine Task
CD	Abfragen und Setzen des aktuellen Directorys
COPY	Kopieren von Dateien oder Directories
DATE	Abfragen und Setzen von Datum und Zeit
DELETE	Löschen von Files oder Directories
DIR	Directory anzeigen und bearbeiten
DISKCOPY	Kopieren ganzer Disketten
ECHO	Ausgabe einer Meldung nach einem Auftrag
ED	Komfortabler Fullscreen-Editor
EDIT	Editor für zeilenweise aufgebaute Files
ENDCLI	Schließen des CLI-Fensters
EXECUTE	Ausführung einer Kommando-Datei mit DOS-Befehlen
FAILAT	Fehlerfalle für Kommando Dateien
FAULT	Ausgabe von Klartext-Fehlermeldungen
FILENOTE	Anlegen eines Kommentars zu einem File
FORMAT	Formatieren einer Diskette
IF	Entscheidung in Kommando-Dateien
INFO	Ausgeben einer Übersicht über das File-System
INSTALL	Erzeugen einer Boot-Disk
JOIN	Zusammenfassen mehrerer Files zu einem neuen File
LAB	Labeldefinition in einer Kommando-Datei
LIST	Ausführliches Directory
MAKEDIR	Einrichten eines neuen Directorys
NEWCLI	Starten eines zweiten CLI-Prozesses
PROMPT	Abfrage und Setzen der Systemmeldung
PROTECT	Schützen eines Files gegen Löschen, Lesen, Schreiben oder Ausführen
QUIT	Ausstieg aus einer Kommando-Datei
RELABEL	Umbenennen einer Diskette
RENAME	Umbenennen einer Datei oder eines Directorys
RUN	Starten eines Hintergrund-Prozesses
SEARCH	Suchen nach Strings in Files
SKIP	Sprung in einer Kommando-Datei
SORT	Sortieren von zeilenweise aufgebauten Dateien
STACK	Abfrage und Setzen der Stackgröße
STATUS	Ausgabe von Informationen über alle existierenden Prozesse
TYPE	Ausdrucken einer Datei als Text oder Hex-Dump
WAIT	Warten auf bestimmte Uhrzeit oder eine festgelegte Zeit lang
WHY	Ausgabe einer Fehlererklärung

Die CLI-Befehle des Amiga

Brückenschlag zu alten Ufern: CP/M-80-Emulation auf dem Atari ST. Wie arbeitet der Computer der Zukunft mit – wenn auch bewährter – Software aus der Vergangenheit?

Rückwärts in die Zukunft

Wenn sich ein kreativer Computerentwickler nach hartem Tagewerk seiner wohlverdienten Frühmorgenruhe hingibt und dann zwei Stunden später aus tiefstem Schlaf schweißgebadet hochschreckt, sind ihm im Traume drei riesige blaue Buchstaben erschienen. Denn seit »Mama Blue – IBM« die Computerwelt mit ihrem »IBM-PC/MS-DOS«-Standard beglückte, erstarren fast alle Computerhersteller wie Kaninchen vor der Schlange. Sie versuchen verzweifelt, sich durch die Produktion sogenannter IBM-kompatibler Personal Computer von den Krümeln zu ernähren, die ihnen die nicht gerade mütterliche Gigantin übrigläßt. Bisher erwies sich jedoch der nicht unbedingt computeravantgardistische IBM-PC mit seiner Pseudo-16-Bit-Struktur als wahre Fortschrittsbremse bei der Hardwareentwicklung. Dabei ist Standardisierung nicht unbedingt das schlimmste aller Übel. Das demonstrierte lange vor IBM die Softwareschmiede Digital Research mit CP/M, für ehrliche 8-Bit-Personal-Computer noch immer das Standardbetriebssystem schlechthin. CP/M ist ein Softwarestandard, dessen durchdachter Aufbau die Verwendung durchaus unterschiedlicher Hardwarekonfigurationen zuläßt. Basierend auf den Mikroprozessoren 8080/8085 und Z80 hielt es als CP/M-80 inzwischen millionenfach in den verschiedensten Anwendungsbereichen Einzug. Diese enorme Verbreitung hatte die wohl umfangreichste Softwarebibliothek aller Zeiten für Anwender und Softwareentwickler zur Folge.

Kaninchen beißt Schlange

Doch die Ära der 8-Bit-Computer neigt sich rapide und sicher ihrem Ende entgegen. Auch wenn es noch immer Hersteller zu geben scheint (darunter auch ein ganz großer und altbekannter aus den Kindertagen der Personal Computer), die die Zeichen der Zeit wohl nicht erkennen wollen. Ein neuer Stern am Mikroprozessorhimmel ist aufgegangen, der lange Zeit unverdientermaßen im Dunkeln schlummerte: der 68000-Prozessor von Motorola. Ein in jeder Hinsicht gewichtiger Veteran der Computerwelt hat ihn endlich aus dem Schlaf geweckt und zum Anführer einer nicht weniger strahlenden Schar von Helfern gemacht. Gemeint ist Jack Tramiel mit der von den Toten auferstandenen Firma Atari und seinem Überraschungscoup Atari ST.

Doch jeder Computer kann seine ganze Leistung nur mit Hilfe von zuverlässiger Software vollbringen. Und um solche Software ist es bei wirklich neuen Computern wie dem Atari ST zum Zeitpunkt ihrer Markteinführung fast zwangsläufig schlecht bestellt. Denn wegen der immer kürzer werdenden Entwicklungszeiten für die Hardware kann es sich kein Computerhersteller leisten, Neuentwicklungen so lange zurückzuhalten, bis Menge und Qualität der verfügbaren Programme mit dem Angebot für bereits etablierte Computer mithalten können. Auch Mister Tramiel und sein ST bilden natürlich darin keine Ausnahme. Wie läßt sich nun ein solches Dilemma lösen, wenn man nicht IBM heißt und der gesamten PC-Welt seinen eigenen Standard aufzwingen kann, aber dennoch ein zukunftweisendes und damit neues Computersystem auf den Markt bringen will?

Atari hat mit dem ST einen Computer geschaffen, der, nicht zuletzt aufgrund seines großzügig bemessenen RAM-Speichers, zumindest theoretisch dafür geeignet ist, unter jedem geläufigen Betriebssystem für Mikrocomputer zu arbeiten. Augenblicklich sind für den 68000-Prozessor die Standardbetriebssysteme OS-9, Unix und CP/M-68 auf dem Markt. Jack Tramiels Entwicklungsingenieure haben nach eigenen Verlautbarungen für die nahe Zukunft OS-9- und Unix- Anpassungen angekündigt. CP/M-68 ist der Kern des gegenwärtigen Betriebssystems TOS. In der Grundstruktur dem BIOS (Basic Input/Output System), BDOS (Basic Disk Operating System) und CCP (Command Control Prozessor) sehr ähnlich, wurde TOS um einige Funktionen des MS-DOS erweitert. Dennoch ist es nicht möglich, CP/M-68-Software ohne Anpassung lauffähig zu machen. Also auch hier kein direkter Ausweg aus dem Softwaredilemma!

Muß man sich letztendlich doch gedulden, bis die »arteigene« Software entwickelt ist? Nein, ein wirklich erfindungsreicher Programmierer gibt so schnell nicht auf. Es gibt nämlich auf Minicomputern und noch größeren Systemen Programme, die ihre kleineren Geschwister, die Mikros, in allen Funktionen simulieren können. In vielen Softwarehäusern werden die Mikrocomputerprogramme auf derartigen Systemen entwickelt. Die erwähnten Simulationsprogramme nennt man Emulatoren. Die Funktion eines solchen Emulators ist leicht zu beschreiben: Er ahmt einen bestimmten Computer und sein Betriebssystem in allen Funktionen nach. Er kann also beispielsweise einen Minicomputer in einen Atari ST verwandeln. Dabei soll diese Emulation in »Echtzeit« erfolgen. Das heißt, der emulierte Atari ST muß die zu entwickelnden Programme in der gleichen Geschwindigkeit ablaufen lassen wie der entsprechende Hardwarecomputer. Könnte man dieses Prinzip auch auf einen 68000-Mikrocomputer anwenden und zum Beispiel auf dem Atari ST einen MS-DOS- oder einen CP/M-Computer emulieren, wäre Software in Hülle und Fülle vorhanden.

Das Tempo macht's

Echtzeitemulation ist nur dann möglich, wenn der emulierende Computer schneller arbeitet als der emulierte Computer. Die derzeitigen 68000-Systeme werden mit Taktfrequenzen bis zu zwölf Megahertz betrieben (der ST mit genau acht Megahertz), der IBM-PC mit knapp fünf Megahertz, und ein üblicher CP/M-80-Computer arbeitet mit vier Megahertz. Computer mit dem 68000-Prozessor sind also potentiell schneller. Zur weiteren Leistungs- und Geschwindigkeitssteigerung trägt außerdem die Tatsache bei, daß der 68000-Prozessor über einen echten 16-Bit-Datenbus verfügt und die Daten intern in 32 Bit breiten Registern verarbeitet. Daher sollten Computersysteme mit diesem Prozessor durchaus in der Lage sein, IBM-PCs oder CP/M-Computer zu emulieren. Die Emulation eines IBM-PC dürfte dabei die schwierigere Aufgabe sein. Der von Commodore für seinen Amiga (vielleicht doch etwas zu optimistisch) angekündigte MS-DOS-Emulator auf Softwarebasis verlangsamt die Abarbeitung von entsprechenden Programmen derartig, daß von Echtzeitemulation kaum noch die Rede sein kann. Ein ebenfalls angekündigter

Hardwareemulator könnte zwar schnell genug arbeiten, würde aber den Preis eines solchen Systems in unvertretbare Höhen katapultieren.

Bei Atari wurde eine neuartige abweichende Lösung ins Auge gefaßt. Entwicklungsziel ist nicht die Simulation eines IBM-PC, sondern eine Art Ubersetzer, der MS-DOS-Programme so umwandelt, daß die »Übersetzungen« anschließend wie andere TOS-Programme auf einem normalen Atari ST ablaufen können. Ob diesen lobenswerten Bemühungen Erfolg beschieden sein wird, muß die Zukunft zeigen.

Keine Notlösung

In der Gegenwart obliegt es also den CP/M-Programmen, die geschilderten Softwareprobleme neuer 16-Bit-Computer zu mildern. CP/M-Programme (wie generell alle Computerprogramme) setzen sich aus einer Folge von Anweisungen zusammen, die der Mikroprozessor eines CP/M-Computers in der Reihenfolge ihres Eintreffens ausführt. Die Aufgabe eines CP/M-80-Emulators besteht dann »nur« darin, die in der Sprache des Z80-Prozessors abgefaßten Befehle so zu übersetzen, daß ein 68000-Prozessor sie verstehen und in entsprechende Aktionen umsetzen kann. Bedingt durch die Forderung nach Echtzeitverarbeitung muß dieser Übersetzungsprozeß sozusagen simultan ablaufen. Im Idealfall sollen weder die Programme noch der Anwender merken, daß hier kein echter CP/M-80-Computer am Werk ist, sondern ein zurechtgestutzter 68000er. Nach soviel theoretischer Überlegung über Aufgabe und Wirkungsweise von Emulatoren muß endlich die Frage nach ihrer praktischen Verwirklichung gestellt werden. Es gibt nämlich bereits einen funktionsfähigen CP/M-80-Emulator, der ursprünglich für das Betriebssystem CP/M-68 geschrieben worden ist. Er konnte ohne allzu große Schwierigkeiten an TOS angepaßt werden. Damit steht dem Atari ST eine wahrhaft gigantische Softwarebibliothek zur freien Verfügung. Der Eintritt in die Bibliotheksräume ist zunächst einmal sehr preiswert, da Atari den Emulator zur kostenlosen Verteilung freigegeben hat. Pünktlich zu Weihnachten 1985 kamen über das Händlernetz neben dem eigentlichen Emulatorprogramm drei Hilfsprogramme in Umlauf. Es handelt sich dabei um ein Kopierprogramm für Systeme mit nur einem Diskettenlaufwerk und je ein Programm zur Dateiübertragung zwischen TOS und CP/M in beide Richtungen. Ein geradezu üppiges Weihnachtsgeschenk für die wachsende ST-Gemeinde!

Am Beispiel des Atari ST sollen die Fähigkeiten, aber auch die Grenzen der Emulation eines Mikrocomputers auf einem Mikrocomputer dargelegt werden. Emuliert wird ein 64-KByte-CP/M-Computer mit einem Betriebssystem der Version 2.2. Für dieses Betriebssystem stehen Programme zur Verfügung, die das Herz eines jeden Computerbesitzers schneller schlagen lassen: Textverarbeitung (Wordstar), Datenverwaltung (dBase), Tabellenkalkulation (Multiplan), Datenfernübertragung (Move-it, Kermit), Computersprachen nach Belieben (Assembler, MBasic, Turbo-Pascal, Cobol, Fortran) und vieles, vieles mehr. Durch die in der Computergeschichte bisher einmalige Standardisierung von CP/M sind alle oben genannten Programme mit dem Emulator auch auf dem Atari ST einsetzbar Nach dem Starten des Programms »CPMZ80.TOS« durch Doppelklick mit der Maus (dies ist übrigens die letzte Tat der Maus, bis zu dem Moment, in dem der Atari ST wieder aus den Niederungen eines 8-Bit-Daseins auftaucht) verschwindet das Desktop und der weiße TOS-Bildschirm erscheint. Kommt man der freundlichen Aufforderung nach und wechselt die TOS-Diskette gegen die CP/M-Diskette mit den Hilfsprogrammen, meldet sich der emulierte CP/M-Computer alias Atari ST mit dem obligatorischen Buchstaben zur Kennzeichnung des aktiven Diskettenlaufwerks, gefolgt von einem Doppelpunkt. Von nun an läßt sich der ST wie ein gewöhnlicher CP/M-Computer behandeln. Die Eingabe des Befehls »DIR« wird mit der Anzeige des Directories der eingelegten Diskette quittiert. Die Directoryanzeige ist erweitert und enthält neben der Angabe der einzelnen Dateigrößen auch Informationen über die Anzahl der gespeicherten Dateien und den auf der Diskette noch freien Speicherplatz. Der Emulator arbeitet nicht nur mit den beiden Diskettenlaufwerken, sondern sogar mit RAM-Disks, die vor Starten des Emulators unter GEM oder TOS initialisiert worden sind.

Hinein ins Emulatorglück

Anfangs müssen sie vor Benutzung durch den Emulator mit dem Befehl »ERA *.*« gelöscht werden. Danach lassen sie sich wie normale Diskettenlaufwerke ansprechen. Selbst Diskutilities wie »DU.COM« registrieren nicht, daß sie eine Pseudodiskette bearbeiten. Man darf sich aber hinsichtlich einer Geschwindigkeitssteigerung von Lade- und Speichervorgängen durch die RAM-Disk nicht zuviel versprechen. Wie Tabelle 1 zeigt, beträgt der Zeitgewinn aber immerhin etwa 30 Prozent. Bei der weiteren Erprobung stellten sich sogar einige Erweiterungen des CCP (Command Control Prozessor = Benutzerschnittstelle) heraus: »ERA« listet die Namen der gelöschten Dateien auf, der Befehl »GO« startet erneut ohne Nachladen ein Programm in der TPA (Transient Programm Area, Programmspeicher für Anwenderprogramme), »EXIT« bewirkt einen Rücksprung in den GEM-Desktop. Erfreulich, daß der wahre ST sein Dasein nicht völlig verleugnet. Trotz Emulation kann man jederzeit durch Drücken der Tasten »ALTERNATE« und »HELP« vom Drucker eine Bildschirm-Hardcopy anfertigen lassen. Nach Beendigung des Druckvorgangs sind auch CP/M-Programme problemlos fortzusetzen.

Hin und her, gar nicht schwer

Leider können TOS- und CP/M-Disketten nicht im jeweils anderen System gelesen und beschrieben werden. Aufbau und Ort des Directory unterscheiden sich nämlich in beiden Systemen. Disketten mit CP/M-Programmen sollten besonders deutlich beschriftet werden, da sie im GEM-Directory nämlich wie leere, formatierte Disketten erscheinen. Sogar die Zusatzprogramme zum Datentransfer zwischen CP/M und TOS funktionieren einwandfrei. Nachdem der Emulator seine prinzipielle Funktionsfähigkeit erwiesen hat, sollte nun seine Leistungsfähigkeit mit Anwenderprogrammen gemessen werden. Zu diesem Zweck wurden die Ladezeiten mehrerer Standardprogramme und die Ablaufgeschwindigkeiten einiger ihrer Funktionen, im Vergleich mit dem Hardware-CP/M-Com-

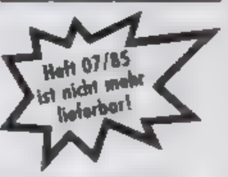
Vergleich der Ladezeiten einiger CP/M-Programme

	Wordstar	Multiplan	dBase II	dBase II Help	Turbo-Pascal
Atari ST mit Emulator	14,6	17,0	5,7	9,9	3,9 sec.
Atari ST mit Emulator und RAM-Disk	12,2	12,2	2,8	7,6	2,4 sec
Kaypro II	9,0	12,0	8,3	8,7	10,1 sec

Tabelle 1. Soft-Z80 gegen Hard-Z80: Leichte Vorteile für die Hardware

Ergänzen Sie jetzt Ihre HAPPY COMPUTER-Sammlung

Schaffen Sie sich ein interessantes Nachschlagewerk und gleichzeitig ein wertvolles Archiv!



Kennen Sie alle »Happy Computer«-Ausgaben von 1985? Suchen Sie einen ganz bestimmten Testbericht? Oder haben Sie einen Teil eines interessanten Kurses versäumt? Suchen Sie nach einer speziellen Anwendung?

Damit Sie jetzt fehlende Hefte mit »Ihrem« Artikel nachbestellen können, finden Sie auf diesen Seiten eine Zusammenstellung aller wesentlichen Artikel der Ausgaben 01 bis 06 und 08 bis 12/85.

Und so kommen Sie schnell an die noch lieferbaren Ausgaben: Prüfen Sie, welche Ausgabe in Ihrer Sammlung noch fehlt, oder welches Thema Sie interessiert. Tragen Sie die Nummer dieser Ausgabe und das Erscheinungsjahr (z.B. 2/85) auf dem Bestellabschnitt der hier eingehefteten Bestell-Zahlkarte ein. Die ausgefüllte Zahlkarte einfach heraustrennen und Rechnungsbetrag beim nächsten Postamt einzahlen. Ihre Bestellung wird nach Zahlungseingang umgehend zur Auslieferung gebracht.

puter Kaypro II untersucht. Das Ergebnis des Vergleiches der Ladezeiten (Tabelle 1) zeigt kein einheitliches Bild. Bei Wordstar und Multiplan hat der Kaypro II die Nase vorn, bei dBase und Turbo-Pascal dagegen gewinnt der ST mit Emulator deutlich an Boden. Compilierung und Ausführungsgeschwindigkeit eines bekannten Benchmarkprogramms (Sieb des Erathostenes) lassen den Atari ST ein wenig ins Hintertreffen geraten (Tabelle 2). Mit Multiplan weiß der Emulator nichts Rechtes anzufangen. Es dauert jeweils geraume Zeit, bis ein eingegebener Text oder eine eingegebene Zahl (für effektives Arbeiten etwas zu langsam) endlich auf dem Bildschirm erscheint. Hier scheint die Grenze einer Emulation erreicht zu sein.

Benchmarktest Turbo-Pascal mit ERATOS.PAS		
	Atari ST mit Emulator	Kaypro II
Compilierzeit	2,3 sec.	1,0 sec.
Laufzeit Compilierung		
1 Durchlauf	11,1 sec.	4 0 sec.
2 Durchläufe	21,7 sec.	7,8 sec.
3 Durchläufe	32,5 sec.	11,6 sec.

Tabelle 2. Trotz Emulator ein echter Turbo

Benchmarktest Wordstar 3.00 mit 275 Zeilen Text (Help-Funktion abgeschaltet)		
	Atari ST mit Emulator	Kaypro II
Eine Zeile (65 Zeichen) mit Cursor überfahren	13,1 sec.	5,7 sec.
Sprung von Textanfang bis Textende mit Bildschirmaufbau	7,6 sec.	3,4 sec.
Bildschirmaufbau	4,3 sec.	2,1 sec.
Suchen des letzten Wortes im Text vom Textbeginn aus	10,1 sec.	4,1 sec.

Tabelle 3. Tempo für WordStar

Wordstar-Anpassung für Atari ST mit CP/M-Emulator			
Label	Adresse	alt	neu
HITE	$0248	$18	$19
WID	$0249	$50	$50
CLEAD1	$024A	$02	$02
	$024B	$1B	$1B
	$024C	$3D	$59
LINOFF	$025E	$20	$00
COLOFF	$025F	$20	$00
ERAEOL	$026D	$00	$02
	$026E	$00	$1B
	$026F	$00	$4B
LINDEL	$0274	$00	$02
	$0275	$00	$1B
	$0276	$00	$4D
LININS	$027B	$00	$02
	$027C	$00	$1B
	$027D	$00	$1B
IVON	$0284	$00	$02
	$0285	$00	$1B
	$0286	$00	$70
IVOFF	$028B	$00	$02
	$028C	$00	$1B
	$028D	$00	$71
DELCUS	$02AE	$0A	$00
DELMIS	$02AF	$05	$00

Tabelle 4. Auch als Emulator-Schreiber ein Star

Wordstar schwarz auf weiß

Der hervorragende Monochrom-Monitor prädestiniert den Atari ST geradezu für die Textverarbeitung. Wer auf sein heißgeliebtes Wordstar nicht verzichten kann, braucht die Anschaffung des Atari ST nun nicht mehr zurückzustellen. Im Vergleich mit dem Kaypro II ist der Emulator zwar immer etwa doppelt so lang am Werk (Tabelle 3), in der praktischen Arbeit mit diesem Textverarbeitungsstandard fällt die reduzierte Arbeitsgeschwindigkeit jedoch kaum ins Gewicht. Allerdings sollte der Bildschirm mit Hilfe des Programms »INSTALL.COM« optimal installiert werden. Tabelle 4 zeigt die Werte für eine richtige Wordstar-Anpassung an den Atari ST mit CP/M-80-Emulation. Viel hinderlicher für effektives Schreiben wirkte sich beim Test aber die Tatsache aus, daß der Original-Emulator weder die Eingabe deutscher Zeichen noch die Cursorsteuerung durch die Cursortasten unterstutzt. Der oben schon erwähnte Softwareentwickler aus München ist auch hier nicht untätig geblieben und hat die entsprechenden Änderungen bereits vorgenommen. Damit steht auf dem Atari ST unter CP/M-Emulation Wordstar uneingeschränkt zur Verfügung. Im Vergleich zwischen dem Software-CP/M-Computer Atari ST mit CP/M-80-Emulator und einem echten CP/M-System schneidet die Emulation erstaunlich gut ab. Multiplan ausgenommen, konnte man mit allen getesteten Programmen einwandfrei arbeiten. Man vergaß dabei manchmal, daß man nicht vor einem echten CP/M-Computer saß. Der Emulator entspricht in etwa einem mit 2 Megahertz getakteten System. Je nach Programm kann sich dieser Wert zum Teil erheblich ändern. Die bisher beobachteten Extremwerte betragen 0,6 und sechs Megahertz. Man kann also mit Fug und Recht von einer gelungenen Echtzeitemulation reden

Die Softwarequelle sprudelt

Nun stellt sich natürlich die Frage, wie man zu diesen CP/M-Superprogrammen kommt. Doch darüber braucht man sich nun wirklich keine Gedanken zu machen, denn die wichtigsten Anwenderprogramme sind inzwischen im Diskettenformat des Atari ST herausgebracht worden, und das zu einem außerordentlich günstigen Preis. Es gibt aber noch einen zweiten Weg in das CP/M-Paradies. Hier sind besonders die Anwender angesprochen, die bereits einen CP/M-Computer mit viel Software und vielen wichtigen und unersetzlichen Dateien besitzen. Dank der großzügigen Schnittstellenausstattung des Atari ST kann man nämlich ohne großen Aufwand durch Rechnerkopplung alle Dateien aus seinem alten

System in das neue übernehmen. Das läßt sich sogar mit recht einfachen Datenkommunikationsprogrammen bewerkstelligen. Sicherer arbeiten aber Standardprogramme wie »Kermit« oder das außerordentlich komfortable »Move-it«. Damit können sogar weit voneinander entfernte Computer über Akustikkoppler/Modem unter Benutzung des Telefonnetzes der Bundespost miteinander in Kontakt gebracht werden. Es ist schon beeindruckend, wenn der eigene Atari ST auf dem Schreibtisch in Köln von einem anderen Computer in München gesteuert wird. Ein kleines Beispiel für die Fähigkeiten des Atari-Emulators und seiner Hilfsprogramme sei hier noch beschrieben.

DFÜ auf Umwegen

Auf einer zum Test nach München geschickten Diskette mit wichtigen Atari ST-Dateien fehlte eine kleine Zusatzdatei. Nun war die Redaktion in München aber zeitlich unter Druck und mußte noch zur selben Stunde mit den Daten arbeiten. Ein auf dem ST unter TOS laufendes Kermit-Programm war nicht verfügbar, wohl aber der CP/M-80-Emulator und »Move-it«. So machte sich ein fantasiebegabter flexibler Redakteur an die Lösung des Problems die sich ein wenig abenteuerlich anhören mag, aber wahr ist: Die Hilfsdatei wurde in Köln unter CP/M-Emulation von der TOS-Diskette auf eine CP/M-Diskette übertragen und mit »Move-it« über die Telefonleitung nach München geschickt. Dort, von der CP/M-Diskette wieder ins TOS-Format gewandelt, konnte der geplagte Redakteur endlich mit den Daten arbeiten. Das Ganze dauerte nur 15 Minuten, kam dafür aber etwas teurer als einen Brief per Diskette zu versenden.

Dieses kleine Beispiel wird zwar in der täglichen Datenverarbeitungspraxis nicht eben häufig vorkommen, zeigt aber klar und deutlich die Leistungsfähigkeit von Computeremulationen auf. Wenn schon die Hin- und Rückwandlung einer Datei ohne Informationsverfälschung abläuft, wieviel einfacher wird dann die Übertragung einer großen dBase-Kundendatei vom altgedienten 8-Bit-CP/M-System auf den neuen 16-Bit-Atari ST sein? Mit dem CP/M-80-Emulator kann man also unbesorgt den Einstieg in die fortschrittliche 16-Bit-Technologie wagen, ohne gleich seinen gesamten Datenbestand in mühevoller Handarbeit neu anlegen zu müssen. Es ist nicht nötig, sich an neue Arbeitsabläufe zu gewöhnen. Man kann die vertraute Software weiter benutzen und ruhig warten, bis echte 16-Bit-Versionen der bekannten Softwarestandards, wie zum Beispiel Wordstar oder dBase, für das ST-System verfügbar sind. Dann werden die alten Daten ohne Umstellungsprobleme mit echter 16-Bit-Geschwindigkeit und ungeahntem 16-Bit-Komfort bearbeitet werden können. Der Weg in diese Zukunft ebnet sich für jeden Computerbesitzer durch den CP/M-80-Emulator in geradezu idealer Weise.

Erfolgreicher Brückenschlag

Beim Brückenschlag zu neuen Ufern der Hard- und Softwareentwicklung kann man sicher sein, daß zumindest ein Brückenpfeiler am alten Ufer auf sicherem Grund steht. Wenn auch die Verfestigung der Brückenfundamente auf der anderen Seite noch etwas Zeit braucht, das Bewußtsein einer festen Basis vertreibt die Angst vor dem Betreten der Brücke.

(W. Fastenrath/aw)

Tabellen, Texte, Dateien: Software für Professionelle

Mögen Computer wie die 68000er technisch noch so ausgefeilt und perfekt sein – ohne passende Programme hat man nicht viel Freude daran. Erst die Software macht sie unschlagbar.

Klammert man Spiele aus, so werden Heim- und Personal Computer hauptsächlich für drei Anwendungsgebiete eingesetzt. Zur Textverarbeitung, zur Dateiverwaltung und für die Tabellenkalkulation. Hat man solche Programme häufig und womöglich beruflich im Einsatz, so kommt es neben der allgemeinen Leistungsstärke des Programms vor allem auch auf eine übersichtliche und unkomplizierte Bedienerführung an. Der Atari ST bringt durch seine komfortable Benutzeroberfläche GEM die besten Voraussetzungen dazu mit.

Nun mag man sich fragen, inwieweit beispielsweise die Maussteuerung ein Textverarbeitungsprogramm komfortabler machen kann; der Großteil der Eingaben, also der zu schreibende Text, erfolgt sowieso über die Tastatur. Textverarbeitungsprogramme wie 1ST_Word offenbaren nun in kurzer Zeit, daß vor allem der Einsatz von GEM-Optionen das Bearbeiten von Texten erleichtert. So bietet beispielsweise 1ST_Word alle Vorteile der Window-Technik. Man kann also einen Text in bekannter Manier mittels der Maus horizontal und vertikal über den Bildschirm scrollen lassen. Da unter GEM maximal vier Fenster gleichzeitig aktiv sind, kann man auch bis zu vier Texte gleichzeitig bearbeiten. Zwischen den verschiedenen Texten lassen sich einzelne Blöcke austauschen oder kopieren. Dies geschieht auch wieder am besten mit der Maus. Man rahmt den zu kopierenden Textabschnitt einfach ein und plaziert ihn in einem Puffer. Dann klickt man das Fenster an, in das der Block hineinkopiert werden soll. Ein Kommando reicht also vollauf aus, den Block dann ab der aktuellen Cursorposition im neuen Text abzulegen.

Unter den eigentlichen Fenstern befindet sich noch ein Kasten, der den kompletten Zeichensatz des ST wiedergibt. (Bild 1). Sei der Text französisch oder spanisch, um Sonderzeichen werden Sie nie verlegen sein. Man klickt das Zeichen einfach mit dem Mauscursor an und schon taucht es an der momentanen Textcursorposition auf.

Auch die Einstellung des rechten Textrandes und der Tabulatoren übernimmt die Maus. Hierfür existiert am oberen Bildrand eine gesonderte Zeile, in der die Tabulatorstops und die Ränder symbolisch dargestellt sind. Man klickt beispielsweise das Symbol für den rechten Rand an und verschiebt es dann an die gewünschte Stelle.

Alle übrigen Kommandos reagieren sowohl auf Aufrufe der Maus über ver-

schiedene Pull-Down-Menüs, oder aber auf die zehn Funktionstasten. Als kleine Gedächtnisstütze für die Tastenbelegung findet man am unteren Bildrand eine symbolische Darstellung der zehn Funktionstasten und ihrer Bedeutung. Diese Symbole können wiederum mit der Maus angeklickt werden.

Zur optischen Hervorhebung von Textteilen dienen bei 1ST_Word verschiedene Schriftarten. Buchstaben lassen sich fett gedruckt, kursiv, unterstrichen oder blaß darstellen. Wie die Sonderzeichen können auch diese Schriftarten natürlich auf dem Drucker ausgegeben werden. Zur Anpassung an verschiedene Druckertypen existiert ein spezielles Installationsprogramm. Hier kann man jedem Zeichen den benötigten Druckercode, bei Grafikzeichen im Bedarfsfall sogar ganze Codesequenzen zuordnen. Ob der Drucker dabei über die parallele oder serielle Schnittstelle angesprochen werden muß, spielt keine Rolle.

Ebenso wie Textverarbeitungsprogramme dienen auch Dateiverwaltungen neben geschäftlichen Anwendungen oft rein privaten Zwecken. Eine der wichtigsten Voraussetzungen, die man an eine gute Dateiverwaltung stellen kann, ist größtmögliche Flexibilität ohne Einbußen an Übersichtlichkeit und gute Bedienerführung. Daneben sind auch schnelle Such- und Sortierroutinen unerläßlich. Was nutzt das aufwendigste Dateiverwaltungsprogramm, wenn man einen gesuchten Eintrag in einem normalen Karteikasten schneller findet.

Von den momentan für den ST erhältlichen Dateiverwaltungsprogrammen bietet sich beispielsweise »DB Master One« an. Es besteht aus zwei Teilprogrammen. Das erste nennt sich »MAKE-ONE«. Wie der Name schon vermuten läßt, handelt es sich hierbei um ein Programm, mit dem man den Aufbau eines Datensatzes und somit die Funktion einer Datei festlegen kann. Man stelle sich den Bildschirm als Teil eines großen Arbeitsblatts vor, auf dem alle Eintragungen zu einer bestimmten Person oder Sache vorgenommen werden sollen. Hierzu legt man sich eine Reihe von Feldern auf dem Arbeitsblatt an, die zur näheren Beschreibung notwendig sind. Baut man also zum Beispiel eine Adressenverwaltung auf, so werden die Felder die Bezeichnungen Name, Vorname, Wohnort, und so weiter erhalten. Die Position und Anordnung und die Größe der einzelnen Felder bestimmt man mit der Maus. Die Bezeichnung eines Feldes wird natürlich mit der Tastatur eingegeben. Die Art der verwendeten Buchstaben wählt man zuvor aus einem Untermenü. So läßt sich die Größe der Schrift, Fett- oder Kursivdruck, sowie unterstrichener Text voreinstellen. Auch die Farbe, in der ein Text erscheinen soll, kann man frei wählen. Allerdings läßt die derzeitig erhältliche englische Version keine deutschen Umlaute zu. Ein weiterer Nachteil der englischen Version macht sich in der Auswahl vordefinierter Feldgruppen bemerkbar (Bild 2). So kann man in einem Menü fertige Felder, wie beispielsweise Adressen, abrufen. Die spezifisch englische Anordnung dieser Felder sowie deren Bezeichnungen nutzen dem deutschen Anwender jedoch wenig. So bleibt zu hoffen, daß in Kürze eine deutsche Version dieses Programms diesen Notstand behebt.

Mit dem zweiten Programm »USE-ONE« kann man schließlich die eigentliche Dateiverwaltung betreiben. Alle Datensätze einer Datei sind ständig im Speicher des ST vorhanden. Diese Methode der Datenverwaltung hat den Vorteil, daß die Zugriffszeiten auf die Einzelinformationen minimal sind. Die maximale Anzahl der zu verwaltenden Datensätze hängt jedoch vom verfügbaren Speicherplatz im RAM ab. Besitzt man einen 520 ST+ braucht man sich darüber kaum Gedanken machen. Selbst beim 260 ST spielt dieses Problem spätestens mit der Lieferung des Betriebssystems auf ROMs nur eine geringe Rolle. Das Programm selbst beschränkt die zulässige Größe einer Datei auf maximal 320 KByte. Jeder Datensatz kann eine Länge bis zu 3000 Zeichen umfassen. Selbst dann, wenn man diese Länge voll ausnutzt, was in der Regel kaum vorkommen wird, lassen sich also immer noch über 1000 Datensätze im Speicher aufnehmen. Wenn man nicht gerade die Warenbestände eines Supermarkts verwalten möchte, sollte dies normalerweise ausreichen.

Datenbank für Supermarkt

Beim Ausdruck gespeicherter Daten kann man natürlich das Druckformat selbst bestimmen. Dabei lassen sich bis zu zehn selbst zusammengestellte Formate speichern und bei Bedarf abrufen. Es können einzelne Felder von Datensätzen oder auch nur bestimmte Datensätze gesucht, sortiert und zu einem Report zusammengefaßt werden.

Hört man Tabellenkalkulationn erscheint wohl als erstes das bekannte Lotus 1-2-3 vor dem geistigen Auge. Dieses Programm gibt es für den ST zwar noch nicht, wohl aber VIP-Professional, das zu Lotus 1-2-3 voll Bedienungs- und Daten-kompatibel ist. Allerdings wird derzeit nur eine Vorabversion von VIP-Professional angeboten. Diese Version ist zwar voll funktionsfähig, es handelt sich jedoch um eine reine Textversion, die ohne die Unterstützung durch GEM auskommen muß. Eine GEM-Version — zwar schon fertig – soll aber erst mit dem Erscheinen der ROMs für den ST ausgeliefert werden. Dann kann man sein altes Programm für 39 Mark umtauschen. Dieses Vorgehen hat einen sehr einfachen Grund. Solange das Betriebssystem in den RAM geladen wird, reicht der verfügbare Speicherplatz des 260 ST einfach nicht aus, um eine GEM-Version zu betreiben. Selbst die Textversion benötigt schon soviel Speicherplatz, daß

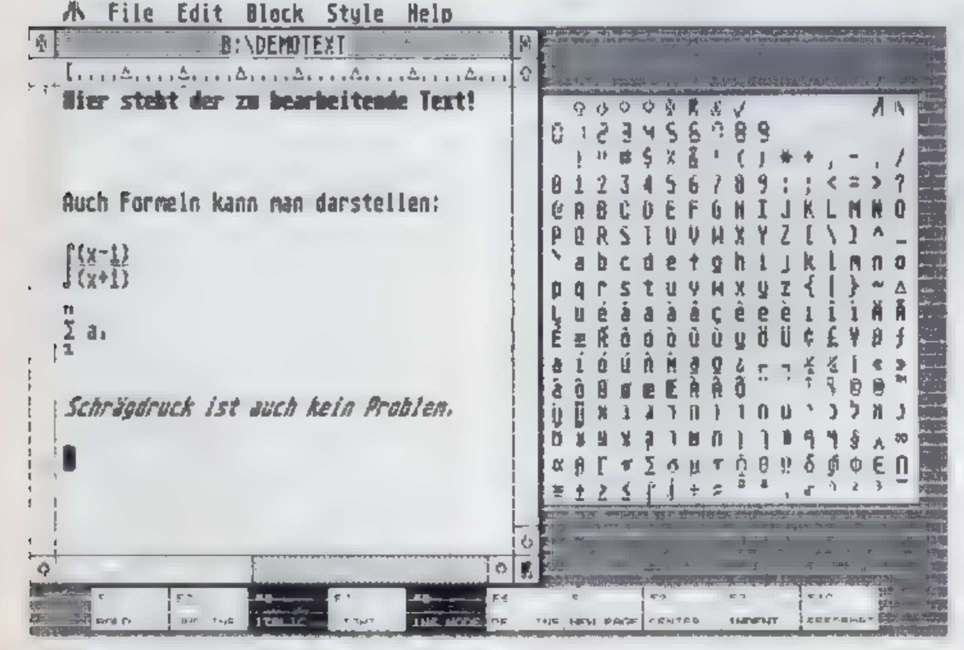


Bild 1. Jedes Symbol aus dem Zeichensatz des Atari ST ist problemlos mit der Maus in den bearbeiteten Text zu übernehmen

Teile des GEMs, speziell das AES, beim Laden von VIP gelöscht werden. Beim Verlassen des Programms kann man daher nicht wie gewohnt zum Desktop zurückkehren, sondern muß neu booten. Dies ist um so ärgerlicher, da VIP-Professional eigentlich aus zwei Teilprogrammen besteht. Das zweite Programm, welches allerdings bereits unter GEM läuft, übernimmt nämlich den Ausdruck von Grafiken. Um es aufrufen zu können, muß man also neu booten und damit das GEM wieder komplettieren.

Sieht man von diesen Nachteilen ab, die mit der Einführung der endgültigen Version sowieso der Vergangenheit angehören, so kann man VIP-Professional als gelungenes Programm bezeichnen. Eine umfangreiche Dokumentation — wenn auch derzeit noch in Englisch — liefert ein rund 250 Seiten starkes Handbuch mit einer Referenzkarte. Auch beinhaltet das Programm selbst zahlreiche Hilfen, die den Einstieg in die Tabellenkalkulation erleichtern. Mit der Taste »/« beispielsweise erreicht man das Hauptmenü, das in einer Zeile am oberen Bildrand untergebracht ist. Fährt man nun mit den Pfeiltasten einzelne Menüpunkte an, so erscheint in der darunterliegenden Zeile jeweils eine kurze Erläuterung der Funktion dieses Menüpunkts. Drückt man die Help-Taste, wird ein umfangreiches Help-Menü eingeblendet, durch das man sich noch weitere Informationen beschaffen kann.

Der Befehlsvorrat von VIP-Professional ist beträchtlich. Ob es nun um mathematische, statistische oder logische Funktionen geht, mit denen man Zellen verknüpfen und berechnen kann, es gibt kaum einen Befehl, den VIP nicht beherrscht. Längere, öfters benötigte Befehlsfolgen können zu Macros zusammengefaßt werden und sind über die Alternate-Taste schnell abrufbar. Die zehn Funktionstasten belegen häufig benötigte Befehle, so daß man zum Teil auch ohne Menüsteuerung zurechtkommt.

Die im Arbeitsblatt gesammelten Daten lassen sich natürlich auch grafisch auf dem Bildschirm darstellen (Bild 3). Verschiedene, frei wählbare Parameter bestimmen das Aussehen der Grafik, des zusätzlichen erklärenden Textes, und so weiter. Jede Grafik läßt sich dann separat abspeichern und mittels des Druckprogramms auf Papier bringen.

Wie auch Lotus 1-2-3 enthält VIP-Professional zusätzlich zur Tabellenkalkulation ein Dateiverwaltungsprogramm. Dies baut natürlich auf der Tabellenkalkulation auf, und läßt sich daher fast genauso handhaben. Jede Zeile des Worksheets beinhaltet dabei

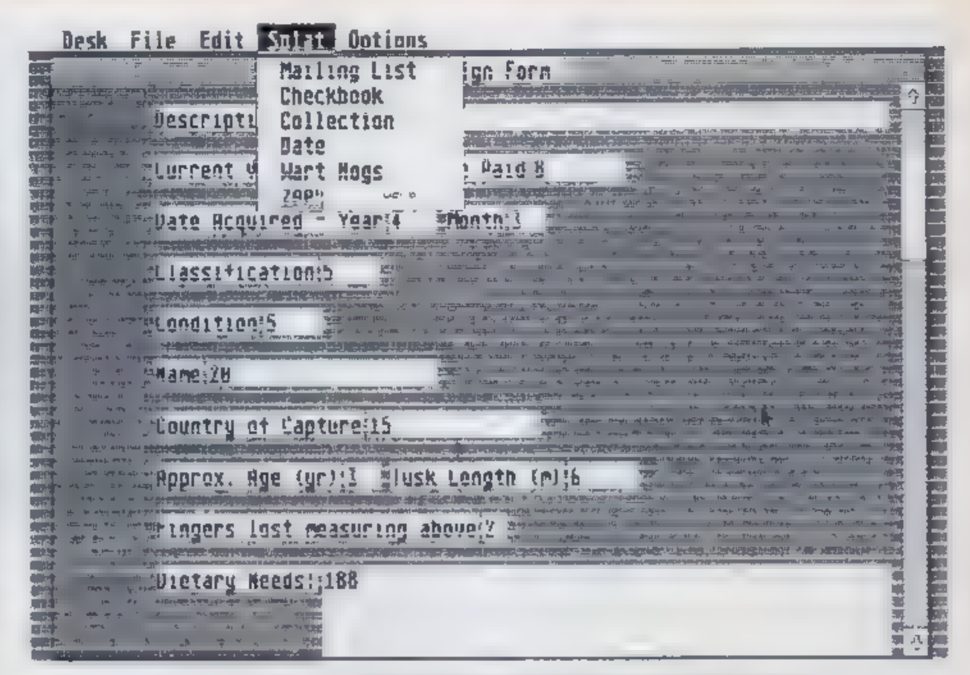


Bild 2. Leider ein Manko des »DB Master one«. Die vordefinierten Felder sind rein auf englische Bezeichnungen ausgerichtet.

zum Beispiel einen Datensatz, wobei das Sortieren, Ändern und Suchen identisch zur Tabellenkalkulation funktioniert. Je nach Speicherplatz kann man Tabellen und somit auch Dateien mit einem Umfang bis zu 256 Spalten und 8192 Zeilen erstellen.

Die wichtigsten Anwendungsprogramme für einen Personal Computer brauchen also einen Vergleich mit der Software von MS-DOS und CP/M-Computern wirklich nicht zu scheuen. Spätestens wenn die Programme, wie es bei VIP-Professional der Fall ist, in ihrer endgültigen Version vorliegen, kann man beim Atari ST vom »echten« Personal Computer sprechen.

Sinclair QL

Der QL wird generell mit vier Programmen ausgeliefert, die von der englischen Firma Psion entwickelt worden sind. Dieses Programmpaket besteht aus einer Textverarbeitung (Quill), einem Datenverwaltungsprogramm (Archive), einem Tabellenkalkulationsprogramm (Abacus) und einem Grafikprogramm (Easel).

Obwohl diese Programme, für sich betrachtet, längst zum Standard jedes Computers gehören, wurden hier einige Aspekte eingebracht, die eine nähere Betrachtung durchaus rechtfertigen. Besonders interessant ist der Datentransfer zwischen den Programmen, die einheitliche Benutzerführung und der pyramidenartige Aufbau.

Voraussetzung für den Datenaustausch innerhalb der Programme ist eine gewisse Konformität. So ist es beispielsweise nicht möglich, grafische Daten in der Tabellenkalkulation darzustellen, obgleich der umgekehrte Fall durchaus denkbar ist. Von Vorteil erweist sich hierbei die Flexibilität der Programme, was die Datenstruktur betrifft. Die Feldlänge eines Eintrages im Datenverwaltungsprogramm ist beispielsweise nicht limitiert, lediglich bei der Bildschirmanzeige wird die Ausgabe formatiert. Dateien, die in andere Programme übernommen werden sollen, sind als sogenannte »EXPORT«-Datei gekennzeichnet, und mit einem »IMPORT«-Befehl kann jederzeit auf sie zugegriffen werden.

Alle Programme weisen optisch ein ähnliches Erscheinungsbild auf. Im oberen Bildschirmteil zeigt eine ausblendbare Kopfleiste die Wahlmöglichkeiten. Die untersten Zeilen stehen für Systeminformationen zur Verfügung, dazwischen liegt der Arbeitsbereich. Innerhalb der Programme werden Funktionstasten, soweit sinnvoll, einheitlich verwendet. So kann zum Beispiel stets mit »F1« ein Overlay-File geladen werden, das Auskunft über die derzeitig zugänglichen Befehle liefert.

Eine wichtige Rolle spielt in diesem Zusammenhang die gestaffelte Programmstruktur. Jedes Programm besitzt mehrere hierarchisch gegliederte Befehlsebenen. Der Anwender hat hier die Möglichkeit, von trivialen bis zu sehr komplexen Aufgabenstellungen die bestmögliche Lösung zu finden.

Auch im Detail bieten diese Programme viele Merkmale, die die Handhabung wesentlich erleichtern.

In der Textverarbeitung wird die Vorlage auf dem Bildschirm ebenso dargestellt, wie sie später ausgedruckt wird. Dies gilt für die Schrift, Hoch- und Tief-

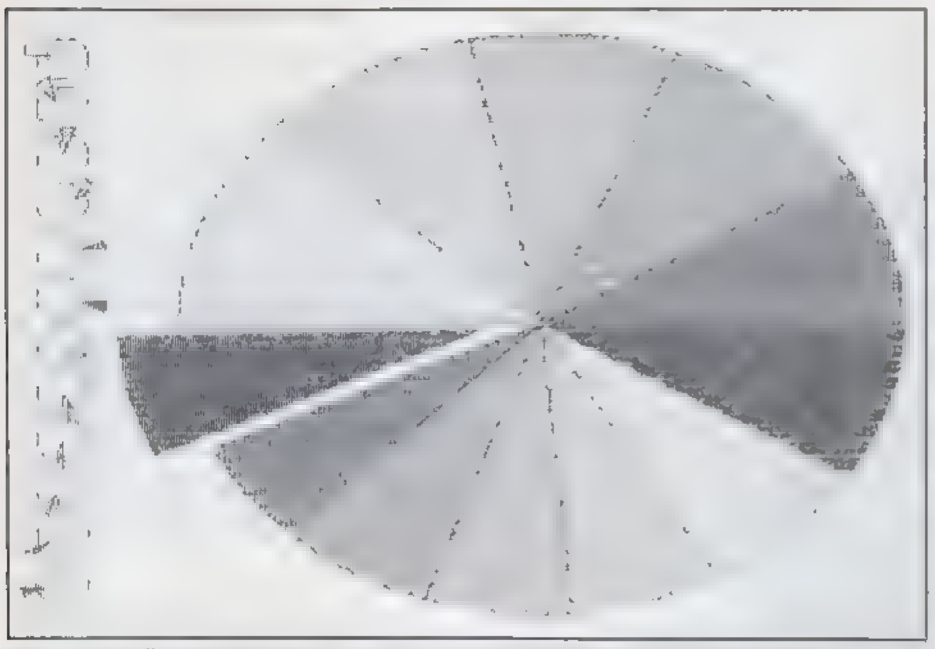


Bild 3. Gute Übersichtlichkeit durch grafische Darstellung: VIP-Professional

stellung, Fettschrift und Unterstreichung ebenso, wie für die Randstellung und die Seitenformatierung. Die Druckeranpassung selbst dürfte ohnehin problemlos vor sich gehen, da mehrere Druckertreiber zur Verfügung stehen.

Das Programm Archive ist, vergleichbar mit dBase II, ein Datenbankprogramm für universellen Einsatz. Leider kommt hierbei nachteilig der Zwiespalt der deutschen Version zum Tragen. Auch die Programmierbefehle müssen Deutsch eingegeben werden, so daß die Analogie mit dem SuperBasic verloren geht. Aus »FOR ... ENDFOR« wird »SOLANGE ... ENDESOLANGE«, etwas Gewöhnung ist dabei schon vonnöten.

Die Arbeitsgeschwindigkeit der Programme, die ursprünglich in C erstellt wurden, ist inzwischen durchaus zufriedenstellend. Verschiedene Programmteile, wie zum Beispiel die relativ langsame Bildschirmausgabe, wurden durch Maschinencode-Routinen ersetzt. Lediglich Zugriffe auf die eingebauten Microdrives stellen die Geduld des Benutzers oftmals auf die Probe. Besonders bei Sortiervorgängen und dem Laden der Hilfsdateien wird die Wartezeit rasch lästig. Zumindest ein Diskettenlaufwerk, besser noch eine Festplatte, sind vonnöten, um der zweifellos sehr guten Software gerecht zu werden.

Und noch ein anderer Punkt macht sich negativ bemerkbar: Für wichtige Daten sind die Cartridges des QL nicht das geeignete Speichermedium. Da das Band an der Hinterseite ungeschützt ist, können durch eine kleine Unachtsamkeit schnell wichtige Daten unlesbar werden und damit verloren gehen.

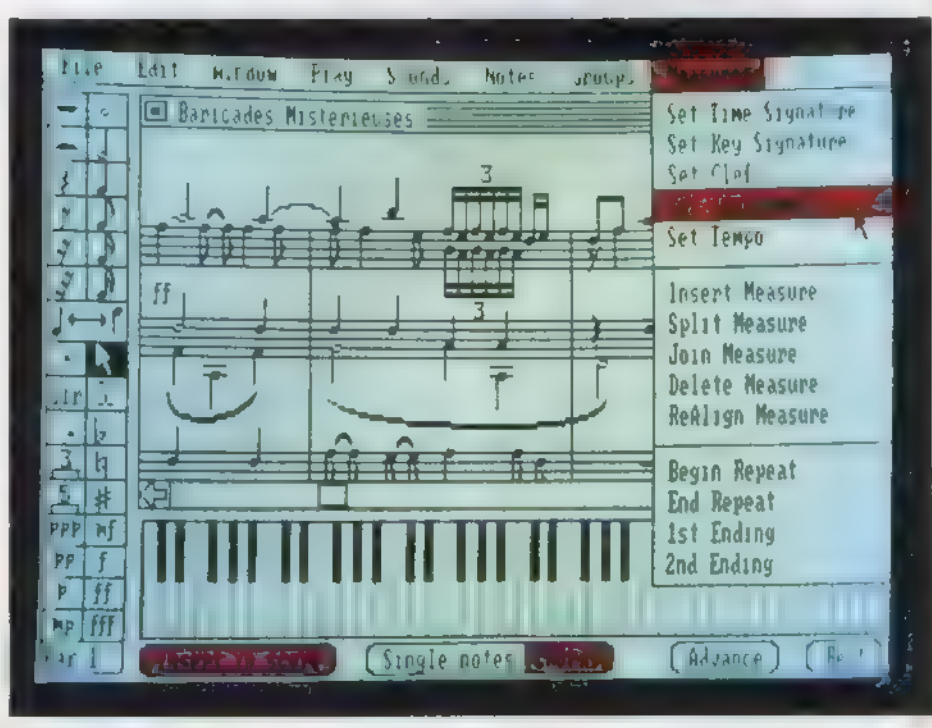
Dennoch setzen diese Programme Maßstäbe, besonders wenn man sie in Relation zu ihrem Kaufpreis und dem Hardwareaufwand setzt.

(Wolfgang Czerny/W. Diekert/hb)

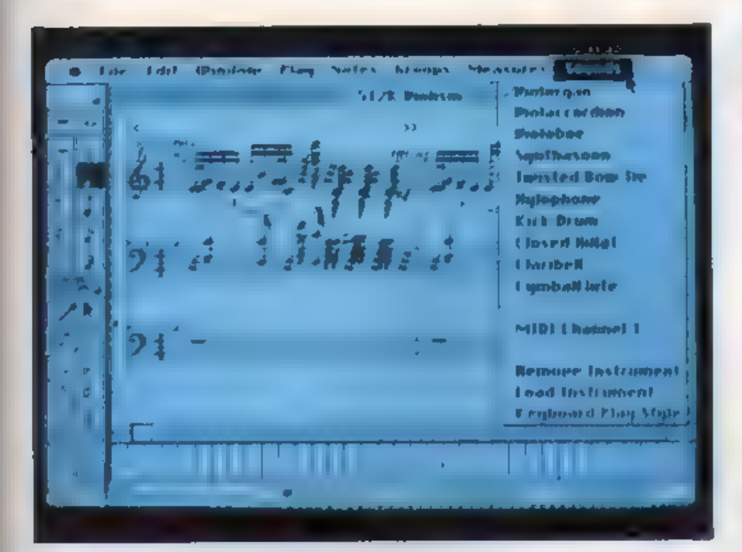
Musik ist Trumpf

Wollten Sie schon immer mal Beethovens Fünfte in einer Jazz-Version hören, mit Saxophon, Posaune und Trompete?
Jetzt werden alle Ihre Klang-Fantasien Wirklichkeit. Das »Deluxe Music Construction Set« macht's möglich!

Musik-Software, die sich hören lassen kann, ist das »Deluxe Music Construction Set«. Wir testeten das neue Musik-Softwarepaket in der Macintosh-Version. Demnächst wird es auch für den Amiga erscheinen, auf dem es durch die hardwaremäßig gegebene Klangdigitalisierung sicherlich noch an Attraktivität gewinnt. Das »Deluxe Music Construction Set« ist auf dem 128 KByte-Mac lauffähig, aber erst beim »Fat-Mac« lassen sich alle Funktionen voll ausreizen. Selbst vierstimmige Arrangements stellen kein Hindernis dar. Sie können in fünf voneinander unabhängigen Windows und zehn Pull-Down-Menüs bearbeitet werden. Am wichtigsten ist das sogenannte Score-Window, in dem die Noten- und Text-Zeilen erscheinen. In diesem Fenster kann man die Noten für bis zu acht Stimmen untereinander einsetzen und sie variieren. Das Note-Palette-Window bietet neben diversen Noten- und Pausenzeichen noch einen Radiergummi und andere »Arbeitswerkzeuge«, während das Piano-Keyboard-Window eine Klaviertastatur beinhaltet. Diese drei



Let the Music play: Das »Deluxe Music Construction Set« auf dem Amiga



Zahlreiche Sounds bestimmen den Klang Ihres Computer-Orchesters

Bildschirmfenster sind die gebräuchlichen Arbeitswindows, auf die man permanent zurückgreift. Daneben existiert noch das Score-Setup-Window, das

Window-Vielfalt

dazu dient, das Erscheinungsbild der Notenzeilen, das Tempo und die Lautstärke sowie das Bild der Notenzeilen beim Ausdrucken zu verändern. Das Memory-Window informiert über den verbleibenden Speicherplatz. Alle Fenster können Sie beliebig auf dem Bildschirm plazieren und das Score-Window sogar in seiner Größe verändern. Im Gegensatz zu vielen anderen Macintosh-Programmen können beim »DMCS« mehrere Windows gleichzeitig aktiv sein. Die Handhabung und Bedienung vereinigt das Prinzip der Construction Sets mit dem typischen Arbeitsprinzip des Macintosh.

Komponieren per Maus

Es gibt zwei Methoden, um »Musik einzugeben«. Entweder holt man mit der Maus eine Note aus dem Note-Palette-Window und setzt sie ins Score-Window ein, oder man klickt ein Kreuz in die Insert-Box im Piano-Window, woraufhin alles, was man auf der Bildschirm-Klaviatur spielt, in Notenschrift im Score-Window erscheint. Die Noten können dann noch ausgeschnitten, kopiert, eingesetzt und gelöscht werden. Noten oder Notengruppen können auch um einen Schritt oder um eine Oktave nach oben oder unten verschoben, Akkorde können umgekehrt und die Dauer von Noten kann verdoppelt oder halbiert werden.

Die Lautstärke bestimmter Stimmen und Partien ist über die Crescendo- und Diminuendo-Kommandos veränderbar. Die Instrumentalbesetzung verschiedener Partien und Stimmen und der Rhythmus (zum Beispiel Staccato, Legato etc.) sind frei variierbar. Sogar die Eingabe von zwei Stimmen auf eine Notenzeile (Klaviermusik) ist kein Problem. Die Beschreibung aller machbaren Manipulationen in allen Variationen würde jedoch den Rahmen dieses Artikels sprengen.

Jede Eingabe wird vom Programm überprüft. So findet zum Beispiel jede neue Note automatisch ihre richtige horizontale Position innerhalb eines Taktes. Auch bleibt es nicht unbemerkt, sollte ein Takt zu viele Noten beinhalten. Nachträglich kann man die Position der Noten mit der Maus noch verändern, sowie ganze Takte dehnen oder kürzen. Ihrer Komposition stehen drei verschiedene Wege zur Wiedergabe offen. Einmal der eingebaute Lautsprecher des Macintosh, der jedoch kein berauschendes Klangbild liefert. Weitaus bessere Ergebnisse sind mit einem externen Lautsprecher/Verstärker (wobei das Programm dann die bessere Dynamik berücksichtigt) oder über ein angeschlossenes MIDI-Interface zu erzielen. Dabei wird die Lautstärke vom Programm aus kontrolliert. Das DMCS arbeitet mit den MIDI-Interfaces »Mini-Mac« von Opcode-Systems und »The Conductor« von Assimilation Process zusammen. Sie werden mit dem Modem-Port des Mac und dem zu steuernden Synthesizer verbunden.

MIDI mischt mit

Wird kein externer Synthesizer verwendet, ist das Programm imstande, 27 verschiedene Instrumente zu simulieren, von denen einige verblüffend realistisch klingen, andere aber voll und ganz der Fantasie des Programmierers Geoff Brown entsprungen sind. Leider kann der Anwender keine eigenen Instrumenten-Klänge definieren. Er muß sich mit dem Vorhandenen begnügen.

Das »Player-Piano«-Kommando bewirkt, daß die Tasten der Klaviatur oder die Noten in den Notenzeilen aufblinken, wenn sie gespielt werden. Die Musikstücke können jederzeit auch mit Textzeilen unterlegt werden.

Zu diesem Zweck lassen sich die Notenzeilen beliebig weit auseinander schieben. Auf der Diskette befinden sich übrigens drei spezielle Zeichensätze, die zur Darstellung von musikalischen Symbolen und Gitarrengriffen dienen.

Spielen, Speichern, Drucken

Alle Kompositionen können natürlich auch gespeichert und ausgedruckt werden. Dabei wird das gesamte Lied inklusive Lautstärke, Tempo, Instrumentierung und Text auf Floppy gebannt. Beim Ausdrucken gilt die Macintosh-Maxime »What you see is what you get« – es wird alles genauso ausgedruckt, wie es auf dem Bildschirm aussieht. Dem Anwender bleibt es also nicht erspart, sich mit komplizierten Formatierungen herumzuschlagen. Das Score-Setup-Window hilft jedoch genau zu definieren, was auf dem Ausdruck erscheinen soll und was nicht.

Kreativität gefragt

Die 50seitige, englische Dokumentation ist sowohl umfassend, als auch leicht verständlich und erklärt die Funktionen des Programms zufriedenstellend. Außerdem enthält sie eine kurze Erklärung wichtiger Begriffe aus den Bereichen Musik und Computer, und hilft bei der Lösung von Problemen, wenn man geschriebene Musik ins Programm übertragen möchte. Aufgrund seiner Benutzerfreundlichkeit und seines klaren Aufbaus ist das »DMCS« für den Nichtmusiker ebenso interessant wie für den Semi-Profi, der die Vielfalt der Gestaltungsmöglichkeiten und die Leistungsfähigkeit des Programms zu schätzen wissen wird. Außerdem macht es sehr viel Spaß, mit Klängen, Stimmen und Noten zu experimentieren und neue Klangwelten durch eine einfach Maus-Bewegung zu erschließen. Durch Maus und Windows ist die Bedienung des Programms spielerisch einfach. Der einzige Wermutstropfen ist, daß man keine eigenen Instrumente definieren kann. Doch abgesehen davon, ist das »DMCS« durchaus in der Lage, selbst einen unmusikalischen Computer-Freak in einen Klangbastler zu verwandeln.

(Thomas Niedermeier/hl)

Info: Ariolasoft, Postfach 7777, 4830 Gütersloh 1

Die neue Spiele-Dimension

Das Zauberwort »68000« bedeutet nicht nur mehr Computer-Power beim Arbeiten und Programmieren, sondern auch mehr Saft und Kraft für starke Spiele. Wir haben einige besonders interessante Titel für Atari ST, Amiga und Macintosh getestet und bewertet.

Computer mit einem 68000-Prozessor sind erst seit relativ kurzer Zeit zu erschwinglichen Preisen erhältlich, doch schon haben sich neben Utilities und Programmiersprachen auch spannende Spiele für diese Super-Kisten in die Händlerregale geschmuggelt. Kein Wunder, denn zumindest in den USA steht der Macintosh in vielen Privathaushalten, aber auch in den Büros ist man (selbstverständlich außerhalb der Arbeitszeit) hin und wieder einem guten Spielchen nicht abgeneigt. Amiga und ST zielen ohnehin auf den Heimbereich. Sie bieten neben hoher Grafikauflösung und massig RAM eine breite Farbenpalette und empfehlen sich so als Spielcomputer der Oberklasse.

In diesem Dreiergespann hat der Macintosh natürlich einen Software-Vorsprung, da er schon länger auf dem Markt ist. Man kann aber davon ausgehen, daß einige Spiele, die momentan nur für den Mac erhältlich sind, in den nächsten Monaten für ST und Amiga umgesetzt werden. Unsere Bewertungs-Kästen machen ersichtlich, auf welchem Computer wir ein Spiel getestet haben. In Klammern stehen außerdem die Computer, für die das Programm noch erhältlich ist.

Der Trend ist positiv: Immer mehr amerikanische Spiele-Hersteller wie Electronic Arts und Epyx haben angefangen, auch Spiele für den Macintosh zu veröffentlichen.

Einige wichtige Firmen halten von Anfang an große Stücke vom Amiga und haben deswegen eine Reihe von Titeln schon in Arbeit. Die Euphorie, die der Atari ST bei uns auslöste, ist in den USA nicht ganz so groß. Doch sind die englischen Spiele-Entwickler als Ausgleich vom preiswerten ST mehr angetan als vom Amiga.

Das Spiele-Angebot wird zwar vorerst nicht so üppig sein wie beim Commodore 64, doch man kann sich ruhigen Gewissens mit dem Schlagwort »Klasse statt Masse« trösten. Einige Glanzlichter aus dem momentanen Angebot haben wir für Sie auf den nächsten Seiten getestet. (hl)

Winter Games

Achtung, Ski- und Rodel-Fans: Der jüngste Sportspiel-Renner von Epyx liegt jetzt schon in einer sehr sehenswerten Macintosh-Version vor. Bei der Olympia-Simulation »Winter Games« dürfen Sie sich in sechs Wintersport-Disziplinen versuchen, die mit Tastatur und Maus gesteuert werden: Skispringen, Bobfahren, Eiskunstlauf (Pflicht und Kür), Trickski und Biathlon. Bis zu acht Spieler können antreten, ihren Namen eintippen und sich eine von 18 Nationen aussuchen, für deren Flagge und Nationalhymne man um olympische Medaillen streitet.

Bei allen Disziplinen sind Geschicklichkeit und etwas Übung gefragt. Bei Trickski und Eiskunstlauf führt man die gewünschten Figuren durch Tastendruck aus, während es beim Biathlon neben einer ruhigen Hand am Schießstand auch auf schnelles Rollen mit der Maus ankommt, um mit Karacho durch die Loipe zu fetzen.

Der Spielwitz des C64-Originals konnte voll erhalten werden. Obwohl man beim Mac auf die Farben verzichten muß, wirkt die Grafik über weite Strecken wesentlich besser. Die sehr gute Auflösung macht die zahlreichen Bilder zum Augenschmaus, und der schnelle Prozessor sorgt bei den animierten Grafiken für Tempo. Besonders schön gelungen sind die graziöse Eiskünstläuferin und die rasante Fahrt des Bobs.

Spielerisch sind die meisten Disziplinen sehr gut geglückt und bringen vor allem mit mehreren Leuten eine Menge Spaß. Kleine Gags, wie die sehenswerte Eröffnungsfeier, das Abspielen der Nationalhymnen und das Speichern der Bestleistungen als Weltrekorde, runden den Spielwitz ab. Nach der überzeugenden Mac-Version will Epyx »Winter Games« vielleicht auch für den Amiga umsetzen. Eine Atari ST-Version ist (noch) nicht im Gespräch. (hl)

Lustvoll durch die Loipe: »Biathlon«, eine der »Winter Games«-Disziplinen

Grafik: Detailreich und sehr schön animiert
Sound: Einige zündende Rhythmen und »echte« Nationalhymnen
Idee: Auffallend einfallsreiche Disziplinen
Motivation: Vor allem mit mehreren Mitspielern sehr hoch
Computer: Macintosh
Preis: zirka 130 Mark
Info: Rushware, An der Gümpgesbrücke 24, 4044 Kaarst 2, Tel. (02101) 68499

Archon

»Archon« gehört bereits seit ein paar Jahren zu den ganz großen Spiele-Klassikern für 8-Bit-Computer. Die neue Amiga-Version ist zwar inhaltlich gleich geblieben, besticht aber vor allem durch Sound-Effekte und eine größere Auswahl an Spiel-Optionen.

»Archon« ist der Kampf zwischen »Licht« und »Dunkel«: Jeder Spieler steuert eines dieser beiden Heere, deren je 16 Figuren über ein quadratisches Spielfeld verteilt sind. Nicht nur diese Spielfläche erinnert sehr an Schach; jede Seite verfügt auch über unterschiedlich starke Figuren, die man strategisch geschickt einsetzen muß.

Gerät eine Figur auf ein bereits besetztes Feld, wandelt sich das Bild zu einem Schlachtfeld. Nun kämpfen die beiden Rivalen um die Herrschaft auf diesem Feld. Bei dieser Sequenz wird das Strategiespiel zum Action-Schlagabtausch, bei dem gute Reflexe gefragt sind. Gesamtsieger ist derjenige, der entweder alle gegnerischen Figuren besiegt, oder fünf bestimmte Felder mit sogenannten »Power Points« besetzt.

Vor allem die Verwendung von digitalisierten Stereo-Sounds sorgt bei der neuen Amiga-Version für die verblüffendsten Effekte. Da zischt ein Pfeil von links nach rechts durch den Bildschirm, oder ein Wasser-Geist blubbert regelrecht über das Spielfeld.

Das »Archon«-Spielbrett und das Optionen-Menü

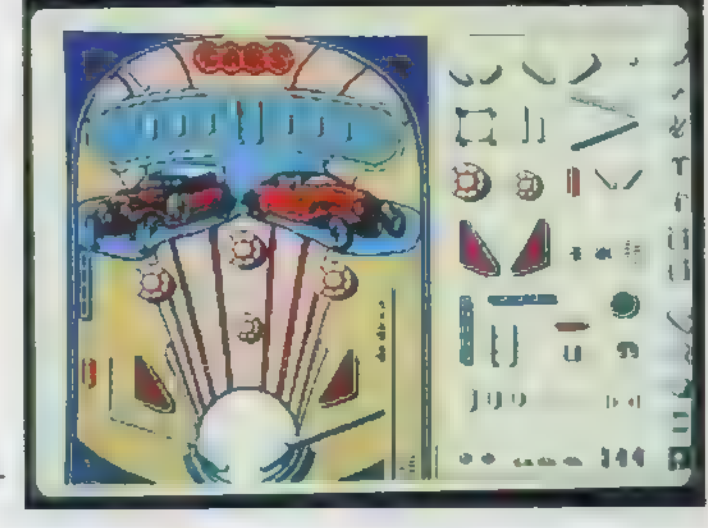
Die Spielvarianten-Palette erlaubt Duelle gegen einen menschlichen Rivalen oder den Computer-Gegner. Letzterer paßt sich in der Schwierigkeit dem Spielkönnen des Menschen an. Außerdem gibt es einen »Cyborg«-Modus für Strategie-Spezialisten, in dem sie die Figuren nur auf dem Brett steuern. Die Action-Sequenzen erledigt der Computer dann im Alleingang. Summa summarum: Ein echter Klassiker im neuen Gewand und sehr empfehlenswert. (hl)

Grafik: Recht gut, nicht allzu aufregend
Sound: Tolle digitalisierte Stereo-Effekte
Idee: Sehr reizvolles Spielprinzip
Motivation: Langfristig sehr hoch
Computer: Amiga
Preis: um die 100 Mark
Info: Ariolasoft, Postfach 7777, 4830 Gütersloh 1, Tel. (05241) 80 51 65

Pinball Construction Set

Am guten alten Flipper kommt man anscheinend auch im Computer-Zeitalter nicht vorbei – wie sonst ließen sich die zahlreichen Flipper-Simulationen für Heimcomputer erklären? Den ungekrönten König dieser Programme gibt es jetzt auch für Mac und Amiga: das »Pinball Construction Set« (PCS). Wie der Name schon andeutet, kann man mit ihm nicht nur auf dem Bildschirm herumflippern, sondern auch kreativ sein und nach Lust und Laune eigene Flipper zusammenstellen. Das geht dank Maus und Menüs leicht von der Hand und dürfte auch denjenigen keine Schwierigkeiten machen, die keine Ahnung vom Programmieren haben.

Damit man sich erst einmal ein bißchen einspielt, sind fünf fertige Flipper bereits auf der Programm-Diskette gespeichert. Darunter befinden sich einige spielerisch sehr ausgefeilte Exemplare wie »Phantom«, bei dem fast alle Flipper-Teile unsichtbar sind. Obwohl sich auch die Grafik sehen lassen kann, sorgen die Sound-Effekte für die meisten »Aahs« und »Oohs«. Die Programmierer haben nämlich die Spielgeräusche von »echten« Flippern digitalisiert und in das PCS übertragen. Wenn der Ball an Bumper und Spinner hämmert, scheppert und rappelt es aus dem Lautsprecher, als würde ein echter Flipper im Wohnzimmer stehen. Aus Speicherplatzgründen funktionieren diese tollen Sounds leider bei der 128 KByte-Version des Macintosh nicht.

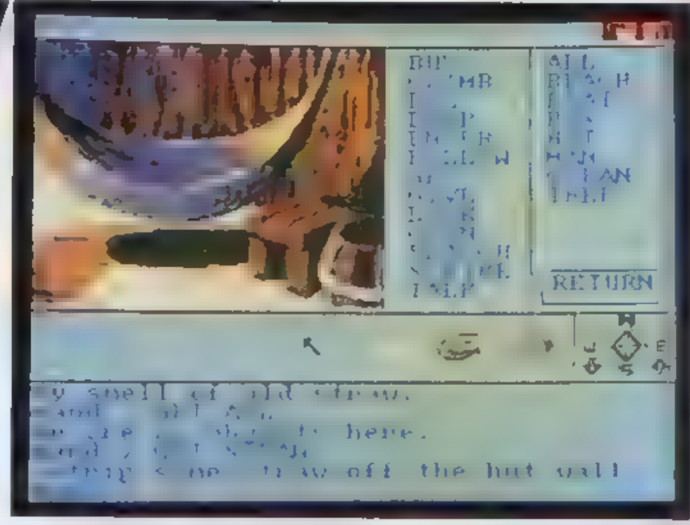


Flippern nach Belieben mit den »PCS«

Zum Spielspaß kommt noch das Construction Set, dessen Auswahl selbst alte Flipper-Hasen befriedigen dürfte. Neben Grafik und Layout können Sie alle wichtigen Spiel-Parameter wie Punktezählung, Ballgeschwindigkeit und Gravitation beeinflussen und mit Sound-Effekten nur so um sich werfen. Die Macintosh-Version ist außerdem MacPaint-kompatibel, damit Sie mit diesem Grafikprogramm Ihrem Flipper den letzten optischen Schliff geben können. (hl)

Grafik: Nutzt die Hardware gut aus
Sound: Fantastisch »echte« Flipper-Geräusche
Idee: Nicht neu, aber gut gemacht
Motivation: Das Nonplusultra für Flipper-Fans
Computer: Macintosh (Amiga)
Preis: um die 100 Mark
Info: Ariolasoft, Postfach 7777, 4830 Gütersloh 1, Tel. (05241) 80 51 65

Mindshadow/ Borrowed Time



Hängematten-Idylle bei »Mindshadow«

»Wer bin ich?« ist die große Frage bei Activisions Erfolgs-Adventure »Mindshadow«, denn Sie erwachen zu Beginn an einem Strand und können sich partout an nichts erinnern: totale Amnesie! Wer Sie sind, woher Sie kommen und wie es Sie auf dieses unwirtliche Eiland verschlagen hat, müssen Sie selbst herausfinden.

Bei den 68000-Umsetzungen wurden die zahlreichen Grafiken natürlich deutlich aufgepäppelt. Der Text des englischen Adventures ist allerdings gleich geblieben; das Sprachverständnis des nur mittelmäßigen Parsers hält sich in Grenzen.

Dafür hat man die Bedienerfreundlichkeit wesentlich verbessert. Die wichtigsten Hauptwörter und Verben kann man aus zwei Auswahlkästen mit der Maus anklicken. So lassen sich bequem eine Reihe von Handlungen ausführen, ohne auf der Tastatur rumhacken zu müssen. Dieser Service erleichtert auch das Weiterkommen in kniffligen Situationen, da man ständig eine Reihe von Wörtern vor Augen hat, die das Programm versteht. Um das Adventure zu lösen, muß man aber auch des öfteren mal »in die Tasten greifen«, da der gesamte Wortschatz des Spiels nicht auf den Bildschirm paßt. Außerdem soll die ganze Sache ja auch nicht zu einfach werden.

Auch Activisions neuestes Adventure »Borrowed Time« hat diese komfortable Benutzerführung und schöne Bilder. Die Handlung ist ein Leckerbissen für Krimi-Fans. Sie schlüpfen in die Rolle eines Privatdetektivs à la Marlowe, der schon so manchen Fall gelöst hat. Leider haben Sie sich damit den Zorn eines Unterwelt-Bosses zugezogen, der von nun an ständig Killer auf Sie hetzt. Ein sehr originelles Abenteuerspiel mit witzigen Texten und Bildern. Besondere Kennzeichen: Trenchcoat und Trommelrevolver. (hl)

Grafik: Viele sehenswerte Bilder
Sound: Keine besonderen Effekte
Idee: Für Adventure-Verhältnisse recht einfallsreich
Motivation: Gut, wenn man Abenteuerspiele mag
Computer: Amiga (Atari ST, Macintosh)
Preis: zirka 80 Mark
Info: Activision, Postfach 76 06 80, 2000 Hamburg 76

Wizardry

»Wizardry«, ein Rollenspiel-Klassiker im neuen Gewand

Das Kult-Rollenspiel »Wizardry« erschien bereits 1981 für die Apple II-Computer und mauserte sich zu einem der größten Verkaufserfolge aller Zeiten. Mit »MacWizardry« ist jetzt die erste Version für einen 68000-Computer erschienen; Umsetzungen für Atari ST und Amiga sind geplant und sollen im Laufe dieses Jahres folgen. Die Entwicklung der Macintosh-Version dauerte ein ganzes Jahr, während dem das ganze Spiel neu programmiert und vor allem grafisch verbessert wurde.

Ihr Computer öffnet Ihnen hier das Tor zu einer anderen Welt, die Magie, mutige Helden und böse Zauberer beherrschen. In diesem Spiel mit dem Untertitel »The Proving Grounds of the Mad Overlord« steuern Sie eine Gruppe Abenteurer, die ausziehen, um den bösen Werdna zu besiegen, ihm ein Amulett zu klauen und es zu ihrem Schutzpatron Trebor zurückzubringen. Sie finden sich nach der Zusammenstellung und Ausrüstung Ihrer Truppe in einem zehn Stockwerke tiefen, unterirdischen Labyrinth wieder. Nach einigen spannenden Begegnungen mit diversen Untieren identifiziert man sich schon richtig mit seinen Spiel-Charakteren und freut sich, wenn sie eine Aufgabe bewältigt haben.

Die Mac-Version wird fast ausschließlich mit der Maus gesteuert; lediglich die Zaubersprüche müssen noch eingetippt werden. Die Windows und Pull-Down-Menüs gestalten das Ganze sehr übersichtlich und benutzerfreundlich.

Die Anleitung zu »Wizardry« ist reich an kleinen Gags. Sowohl Spiel als auch Dokumentation sind neben dem amerikanischen Original auch in deutscher, französischer und japanischer Sprache erschienen, damit es beim Fantasy-Genuß keine Sprachbarrieren gibt.

(Manfred Kohlen/hl)

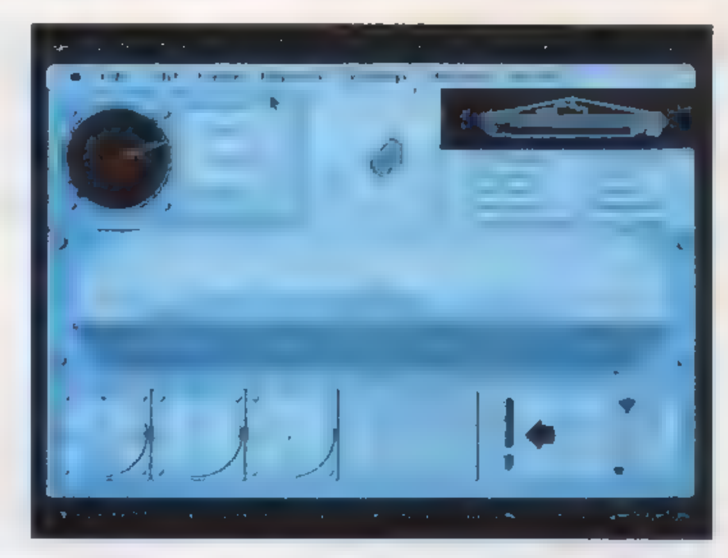
Grafik: Perspektivische Grafik und Windows
Sound: Nur wenige Effekte
Idee: Ein Klassiker, der viele Nachahmer fand
Motivation: Hoch durch eine nicht einfache Aufgabe
Computer: Macintosh (Atari ST, Amiga)
Preis: zirka 200 Mark
Info: Pandasoft, Uhlandstr. 195, 1000 Berlin 12

Gato

In den Tiefen des Meeres... Blubb... Ein U-Boot und seine Mission... Blubb... Sie, der Kommandeur, tragen die Verantwortung... Blubb-Blubb! Die Gewässer sind dieser Tage nicht nur feucht und tief, sondern auch überaus gefährlich, denn die U-Boot-Simulation »Gato« spielt während des 2. Weltkriegs.

Sie steuern ein amerikanisches U-Boot der Gato-Klasse durch den Pazifischen Ozean. Hier können Sie sich für eine von zehn unterschiedlich schwierigen Missionen entscheiden. Mal geht es darum, feindliche Schiffe zu versenken; bei einer anderen Mission ist es Ihre Aufgabe, einen Konvoi der eigenen Truppen zu beschützen. Darüber hinaus kann man sich neue Missionen zusammenstellen – quasi eine Art »U-Boot-Einsatz Construction Set«.

»Gato« ist kein Ballerspiel, sondern eine Simulation, bei der Wert auf Wirklichkeitstreue gelegt wurde. Der Bildschirmaufbau mit den diversen Anzeigen ist sehr übersichtlich. Durch Anklicken mit der Maus gibt man Kommandos oder ruft zusätzliche Informationen ab. Erscheint beispielsweise ein Objekt auf dem Radarschirm, verrät Ihnen das zuständige Besatzungsmitglied, welcher Schiffstyp mit welcher Geschwindigkeit durch die Gewässer kreuzt.



Insel in Sicht: Ein Blick durchs »Gato«-Periskop

Eine Seekarte, in der sämtliche Inseln und Schiffsbewegungen eingezeichnet sind, dient als Orientierungshilfe. Damit das Spiel bei größeren Strecken nicht langweilig wird, laufen in einem »Rapid«-Modus alle Aktionen – auch die der feindlichen Schiffe – zehnmal schneller ab. Ihre seemännischen Erfolge werden zu guter Letzt in einem »Logbuch« gespeichert und bleiben so der Nachwelt erhalten.

»Gato« gefällt durch die spannenden, abwechslungsreichen Missionen, Benutzerfreundlichkeit und Atmosphäre: Um richtig »seefest« zu werden, können Sie sich vom Programm sogar ein paar historische Informationen oder das Morse-Alphabet beibringen lassen. (hl)

Grafik: Klar und übersichtlich mit langsamen 3D-Effekten
Sound: Gleich null, aber unter Wasser ist's nun mal recht ruhig
Idee: Die erste gute U-Boot-Simulation, die sicher Nachahmer finden wird
Motivation: Für Strategie- und Simulations-Fans sehr hoch
Computer: Macintosh (Atari ST)
Preis: zirka 150 (ST) bis 180 Mark (Mac)
Info: Softline, Schwarzwaldstr. 8a, 7602 Oberkirch, Tel. (07802) 3707

One on One

Ganz klar: Ein Sport-Spiel darf in der Software-Palette nicht fehlen. »One on One« ist eine klassische Basketball-Simulation, die in den letzten Jahren schon allerlei Preise und Hitparaden-Plazierungen erringen konnte. Die Versionen für Macintosh und Amiga sind zwar inhaltlich gleich geblieben, setzen aber ihre besseren Grafik- und Sound-Fähigkeiten ein.

Im Gegensatz zu vielen anderen Mannschaftssport-Simulationen scrollt der Bildschirm nicht von links nach rechts. Vielmehr hat man sich auf einen Teil des Spielfelds beschränkt, in dem ein Basketball-Korb hängt. Hier gehen die beiden Spielfiguren gemeinsam mit einem Korb auf Punktejagd. Sie können also nur eine Spielfigur kontrollieren, was für eine »Mannschafts«-Sportart etwas bescheiden ist. Dafür ist die Kontrolle besonders genau und detailreich, was auch für die animierte Grafik gilt.

Abwechselnd dribbelt jeder Spieler mit dem Ball auf den Korb zu. Der Gegenspieler braucht sich nicht aufs Zugucken beschränken, sondern muß versuchen, den Korbwurf abzublocken und den Ball zu ergattern. In einem opulenten Menü lassen sich diverse Parameter wie Länge des Spiels, Stärke des Computergegners etc. einstellen.

Der Basketball-Klassiker »One on One« ist eine Sport-Simulation für Techniker

Die Grafik-Animation ist fließender als bei den 8-Bit-Versionen, und digitalisierte Sounds bringen bei der Amiga-Umsetzung Sporthallen-Atmosphäre ins Wohnzimmer; sei es beim Ballaufspringen, oder dem Quietschen der Turnschuhe. Vor allem Basketball-Freunde werden von der spielerischen Raffinesse sehr angetan sein, die dieses Programm über den Level vieler anderer Sport-Spiele stellt, bei denen nur mit dem Joystick gerüttelt wird. (hl)

Grafik: Ansehnlich animiert, aber nichts Weltbewegendes
Sound: Täuschend echte digitalisierte Effekte
Idee: Originelle Spiel-Idee für Techniker
Motivation: Viele Varianten und daher länger fristig hoch
Computer: Amiga (Macintosh)
Preis: um die 100 Mark
Info: Ariolasoft, Postfach 7777, 4830 Gütersloh 1, Tel. (05241) 80 51 65

Grafikprogramme auf 68000-Computern

Die neuen 16-Bit-Computer bieten sich mit ihren Grafikfähigkeiten für einfache Erstellung von Bildern und Grafiken an. Wir haben einige Zeichen- und Malprogramme unter die Lupe genommen.

Ein hervorstechendes Merkmal der 16-Bit-Computer sind ihre hervorragenden Grafik-Fähigkeiten. Durch Ausstattung mit Spezialchips lassen sich Auflösungen von 320 x 400 Punkten in Farbe und 640 x 400 Punkten in Schwarzweiß einfach erreichen. Natürlich möchte man diese Grafikfähigkeiten dann auch ausnutzen: Für technische Zeichnungen, für Geschäftszeichnungen oder auch nur einfach für bunte Bilder und sogar Kunstwerke. Doch vor das Bild hat man das Programm gesetzt, denn ohne Zeichenprogramm kommt man an die hohe Auflösung nicht vernünftig heran. Die gängigsten Programme für die drei führenden Computer (Atari ST, Amiga und Macintosh) haben wir uns genauer angesehen.

Wichtig ist die Unterscheidung zwischen Mal- und Zeichenprogramm. Ein Malprogramm ist meistens auf farbige Bilder ausgelegt. Hier kommt es weniger auf hohe Auflösung und hochpräzise Zeichnungen an als vielmehr auf möglichst viele Farben. Es soll der Heimanwender und »Künstler« angesprochen werden, der den Computer als Ersatz für den Malkasten ansieht. Außerdem lassen sich bunte Bilder auch für Werbezwecke und ähnliches einsetzen.

Zeichenprogramme hingegen arbeiten meist nur schwarzweiß, nutzen dafür aber die Auflösung des Computers voll aus. Meistens kann man mit Zeichenprogrammen größere Bildflächen bearbeiten, als der Bildschirm überhaupt anzeigen kann. Es sind also immer nur Ausschnitte aus einer Zeichnung zu sehen, die dann dafür in sehr hoher Auflösung komplett als DIN-A4-Blatt auf einem Drucker ausgegeben werden. Als Anwendungsgebiete kommen technische Zeichnungen, Geschäftsberichte und ähnliches in Frage. Manche Zeichenprogramme besitzen zusätzlich noch ein integriertes Textprogramm. Das ergibt eine optimale Kombination von Zeichnung und Text.

Schon beim Kauf des Computers soll man sich die Frage stellen, was man eigentlich will: zeichnen oder malen? Der Macintosh repräsentiert mit seiner reinen Schwarzweiß-Grafik einen typischen Zeichencomputer, der Amiga mit seiner ungeschlagenen Farb-Grafik ein typisches Malgerät. Der Atari ST liegt dazwischen – mit geeigneter Software läßt er sich für beides einsetzen, allerdings nicht ganz so perfekt wie seine beiden spezialisierten Kollegen.

Malprogramme

Beginnen wir mit den Malprogrammen. Für den Atari ST werden augenblicklich zwei Ausführungen angeboten, wobei allerdings noch einiges von anderen Firmen zu erwarten ist.

»Neochrome« (Bild 1) ist ein sehr charakteristisches Malprogramm, dessen Stärke eindeutig in der Ausnutzung der Farbmöglichkeiten des Atari ST liegt. Dafür hat es im Moment noch – es existiert bis jetzt keine endgültige Version – viele andere Schwächen. Im Augenblick ist die Version »0.5« in Umlauf. Bis zu einer endgültigen Version mit einer Eins vor dem Punkt dauert es wohl noch ein Weilchen. »Neochrome« liegt übrigens jedem Atari ST kostenlos bei. Wenn nicht, hat der Händler die Erlaubnis, es Ihnen kostenfrei auf eine leere Diskette zu überspielen.

»Neochrome« ist ein Programm für Farbfanatiker. 16 verschiedene Farben aus 512 möglichen stehen bei einer Auflösung von 320 x 200 Punkten zur Verfügung. Durch ein gut angelegtes Farbmenü kommt man sehr schnell an die einzelnen Farben heran. Durch sogenanntes Color-Cycling, das zyklische Wechseln von Farben, lassen sich auch einfache Animations-Effekte erzeugen. Dafür fehlen viele andere wichtige Funktionen. Man vermißt Kreise, Ellipsen oder Kreisbögen sowie die Auswahlmöglichkeit eines Füllmusters. Aber einem geschenkten Gaul schaut man ja nicht ins Maul, und zum Anfreunden mit den Grafikmöglichkeiten des ST ist »Neochrome« wirklich ideal. Außerdem weiß ja niemand, was in der endgültigen Version noch so alles drinstecken wird.

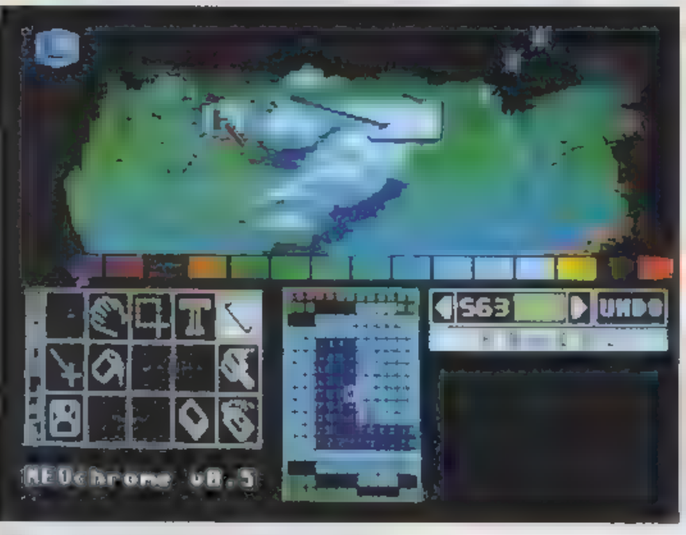


Bild 1. Neochrome für den Atari ST

Bild 2. Degas-Menü für den Atari ST

Das andere Programm namens »Degas« muß man für knapp 160 Mark extra kaufen. »Degas« arbeitet im Gegensatz zu »Neochrome« in allen Grafik-Modi des ST. Man kann also auch vierfarbige Grafiken bei einer Auflösung von 640 x 200 und schwarzweiße mit 640 x 400 Punkten erstellen. Dieses Programm bietet schon sehr viel mehr Funktionen als »Neochrome«, für den verwöhnten Anwender aber immer noch zuwenig. Aus dem ST läßt sich sicherlich einiges mehr herausholen.

»Degas« enthält alle wichtigen Grundfunktionen, wie Linien, Kreise und Vielecke. Interessant ist die Text-Funktion, mit der man Texte in Grafiken einfügen kann. Vier verschiedene Zeichensätze werden dafür mitgeliefert; zusammen mit einem Editor, der eigene Zeichensätze für die Text-Funktion erzeugt. Auch lassen sich das Füllmuster und die Pinselform editieren und auf Diskette speichern. Leider setzt das Füllmuster den Fill-Befehl, der geschlossene Flächen ausfüllt, voraus. Andere Programme (so zu Beispiel MacPaint) können das Füllmuster auch als Pinsel oder Sprühdose verwenden. »Degas« arbeitet mit zwei Bildschirmen, einem für das Menü (Bild 2) und einem für das gezeichnete Bild (Bild 3). Die Hin- und Herschalterei erweist sich dabei als sehr lästig. Insgesamt darf man »Degas« trotzdem als das beste Malprogramm für den ST bezeichnen. Es bietet doch deutlich mehr als »Neochrome«. Die Zukunft beschert aber sicherlich noch bessere Programme, der ST-Softwaremarkt ist ja noch jung.

Bild 3. Ein Beispielbild von Degas

Bild 4. DeLuxe Paint für den Amiga

Besonders bunt mit dem Amiga

Wenn es um bunte Bilder geht, liegt der Amiga klar in Führung. Und gleichzeitig mit dem Erscheinen des Computers gibt es schon zwei Malprogramme, die eigentlich kaum Wünsche offenlassen. »DeLuxe Paint« (Bild 4) und »Aegis Images« (Bild 5) überzeugen durch enorme Funktionsvielfalt. Beide unterscheiden sich in ihrer Leistung nur geringfügig, wobei »Deluxe Paint« durch leichtere Bedienung und schnellere Abarbeitung einige leichte Vorteile verbucht. Da beide das gleiche Format zum Speichern der Bilder auf Diskette verwenden, gestaltet sich der Datenaustausch völlig problemlos. Da wir einerseits »DeLuxe Paint« schon länger kennen und es andererseits auch sehr oft verwenden, berichten wir im folgenden über speziell dieses Programm. Unterscheidungen zu »Images« betreffen ohnehin nur Details.

»DeLuxe Paint« arbeitet in allen Grafik-Modi des Amiga: 320 x 200 Punkte mit 32 Farben, 640 x 200 Punkte mit 16 Farben und 640 x 400 Punkte mit 16 Farben. Der letzte Modus, der Interlace-Modus, fällt durch starkes Flimmern auf, weswegen man ihn nur selten verwenden sollte. Die Farben kann man aus einer Palette von 4096 Farben auswählen, so daß feinste Schattierungseffekte möglich sind. Dies nutzen die Programme denn auch aus: »DeLuxe Paint« kennt allein drei

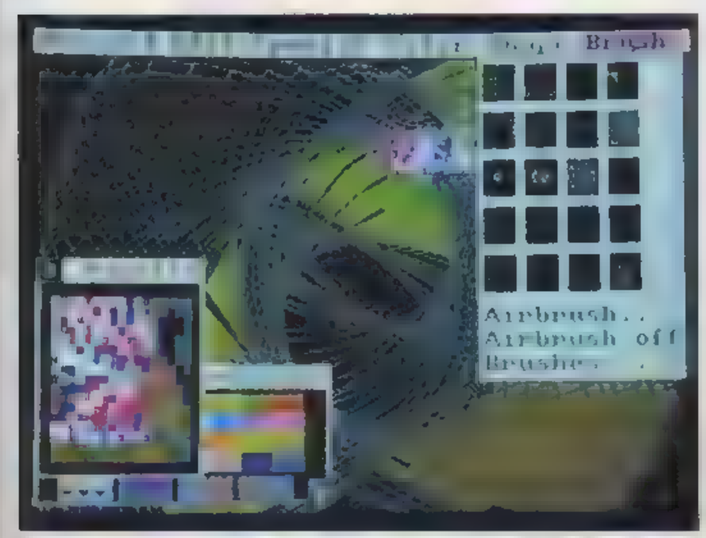


Bild 5. Aegis Images für den Amiga

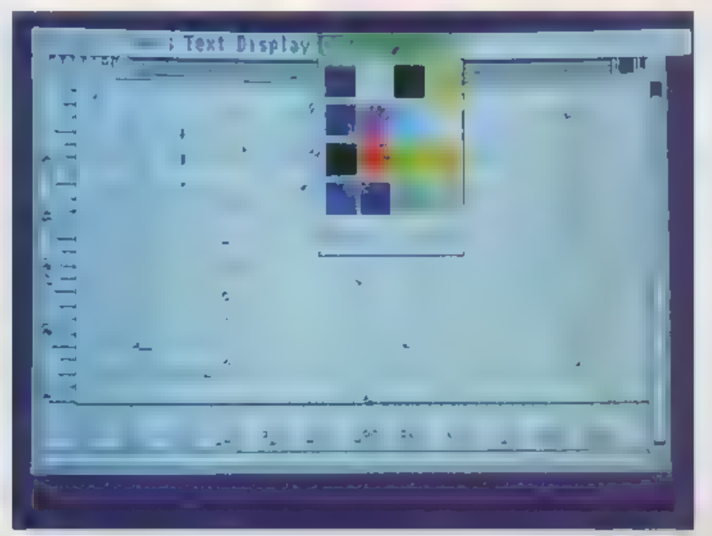


Bild 6. Aegis Draw für den Amiga

verschiedene, zusätzliche Zeichenmodi, die für weiche Farbübergänge, Schattierungen und 3D-Effekte sorgen. Auch hier kann man mit Color-Cycling arbeiten, um einfache Animationen zu erzeugen. Das Malen mit einem Pinsel, dessen Farbe sich zyklisch ändert, ruft so schillernde Regenbogeneffekte hervor. Ganz außergewöhnlich ist die Verwendung von Pinseln ohnehin: Jeder Bildausschnitt, egal wie groß, läßt sich als Pinsel benutzen. Zusätzlich kann man die Größe des Pinsels ändern, ihn verzerren etc. Allein durch diese Option ergeben sich Möglichkeiten wie nie zuvor bei einem Malprogramm. Andere Pluspunkte sind stufenloses »Zooming«, um Bildteile zu vergrößern und wie unter einer Lupe zu bearbeiten, oder einstellbare Symmetrie zum Zeichnen von Rosetten oder schönen Mustern. Alle Vorteile und Fähigkeiten von »DeLuxe Paint« und »Aegis Images« aufzuzählen, würde alleine ein Sonderheft füllen. Mit diesen Programmen, so darf man ungestraft schreiben, eröffnet sich eine neue Dimension der Computer-Kunst, wobei man fairerweise sagen muß, daß dies auch der Amiga mit seinen Grafikfähigkeiten begünstigt.

Zeichenprogramme

Obwohl sich die vorgestellten Programme durchaus professionell nutzen lassen, bieten sie doch für technische Anwendungen zuwenig. Größter Minuspunkt ist die Beschränkung auf eine Bildschirmseite – technische Zeichnungen beanspruchen meist zumindest ein DIN-A4-Blatt. Deswegen gibt es neben Malprogrammen auch Zeichenprogramme. Eines davon ist »Aegis Draw« (Bild 6) für den Amiga. Dieses Programm arbeitet mit einer Auflösung von 640 x 200 Bildschirmpunkten. Dafür ist die tatsächliche Größe der zu bearbeitenden Zeichnung praktisch freigestellt, solange das Ausgabegerät (Drucker oder Plotter) fähig ist, die Zeichnung zu Papier zu bringen. Der Bildschirm gibt immer nur Auschnitte aus der Gesamtzeichnung oder eine Grobübersicht wieder, da man in die Zeichnung hinein- und wieder herauszoomen kann.

Mit 512 x 342 Punkten arbeitet »MacPaint« für den Macintosh (Bild 7). Dieses Programm liegt jedem Gerät umsonst bei. Zusätzlich von sehr hoher Qualität, konnte sich kein anderes Zeichenprogramm für den Macintosh durchsetzen. Das geht sogar soweit, daß viele andere Macintosh-Anwenderprogramme »MacPaint« als Arbeitsgrundlage benützen.

Mit »MacPaint« wird grundsätzlich eine DIN-A4-Seite bearbeitet, von der man auch nur einen Ausschnitt oder eine Grobübersicht auf dem Bildschirm vor Augen hat. »MacPaint« wird eigentlich als Standard unter den Zeichenprogrammen angesehen, da es das erste seiner Art unter den 16-Bit-Computern war. So bietet es auch alle Funktionen, die man heutzutage von einem Zeichenprogramm erwartet. Überragend ist immer noch die Arbeitsgeschwindigkeit von »MacPaint«. Kein anderes Programm ist so schnell beim Linienziehen oder Flächenfüllen. Die einfache Bedienung und Zusammenarbeit mit der Textverarbeitung »MacWrite« tut einiges dazu. So sind schon ganze Bücher auf dem Macintosh entstanden und mit einem Hochleistungsdrucker ausgedruckt und normal vervielfältigt worden.

An diesen Standard anknüpfen zu wollen scheint »GemDraw« (Bild 8). Dieses Programm läuft unter allen Computern, die das Betriebssystem GEM verwenden, wobei der Atari ST die interessanteste Maschine ist. Mit »Gem Draw« kann man ebenfalls beliebig große Vorlagen bearbeiten. Dies wird hauptsächlich dadurch erreicht, daß nicht die Zeichnung, sondern die Zeichenvorschrift gespeichert wird. Dies bringt aber einen entscheidenden Nachteil mit sich: Die Verarbeitungsgeschwindigkeit vermindert sich rapide, da bei Bildwechseln und -verschiebungen der Bildschirm komplett neu aufgebaut werden muß. Ein Vorteil dagegen liegt im Konzept des »Grouping«. Hat man beispielsweise eine Kamera aus verschiedenen Grundelementen (Linie, Vieleck, Kreis) gezeichnet, kann man diese als neues Zeichenelement benutzen und verzerren.

Das letzte Wort ist noch nicht gesprochen

»GemDraw« ist noch mehr an technische Zeichnungen angelehnt als »MacPaint«. So steht die Wahl zwischen verschiedenen Papierformaten frei, man kann Koordinatennetze einblenden und vieles andere mehr.

Wer sich jetzt schon auf »GemDraw« freut, den müssen wir leider enttäuschen. Im Augenblick ist noch nicht geklärt, wie, wann und wo es auf den Markt kommen wird.

Daß auf dem Gebiet Grafikprogramme noch nicht das letzte Wort gesprochen ist, zeigen Ankündigungen verschiedener Firmen. Gerade auf dem Atari ST wird noch fleißig weiterentwickelt. Sicher ist auf jeden Fall, daß es mit einem 68000-Computer und der entsprechenden Software auch für Computer-Neulinge sehr einfach ist, eindrucksvolle Grafiken und Bilder zu zeichnen. (bs)

Info: Neochrome gibt es laut Atari umsonst bei jedem Atari-Händler
Degas: Ariolasoft, Postfach 1350, 4830 Gütersloh 1, zirka 159 Mark
DeLuxe Paint: Ariolasoft, Postfach 1350, 4830 Gütersloh 1, ein deutscher Preis steht noch nicht fest
Aegis Images, Aegis Draw: Aegis Development, 2210 Wilshire 277, Santa Monica, CA 90403, auch hier steht noch kein Preis fest.

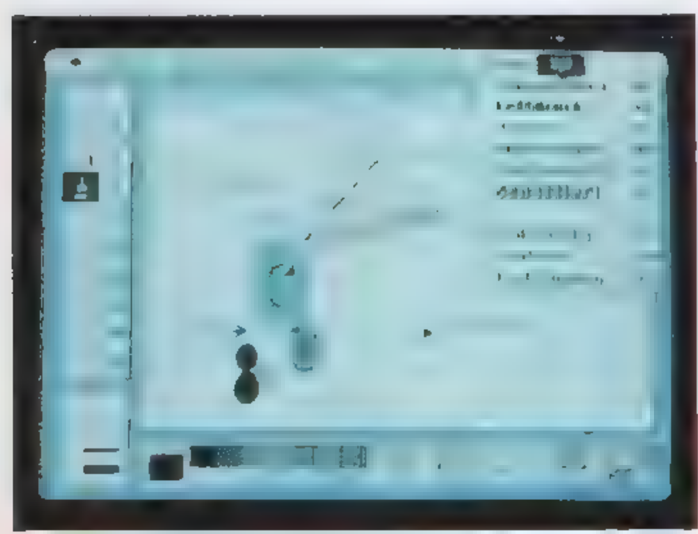


Bild 7. MacPaint für den Macintosh

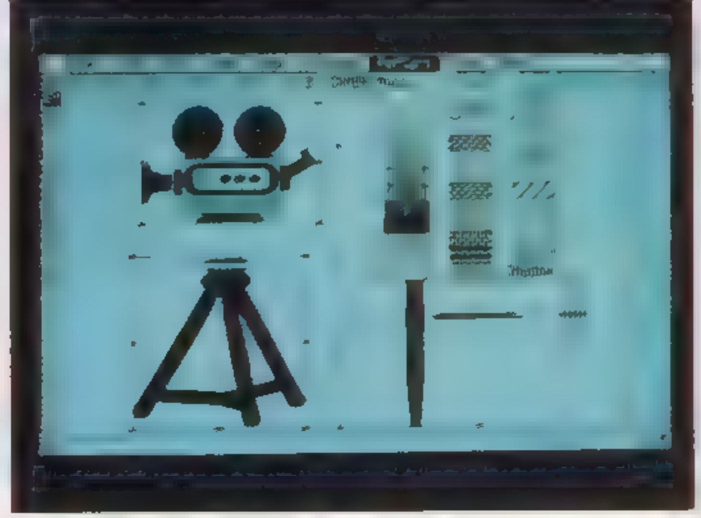


Bild 8. GemDraw für den Atari ST

Puzzlen mit GEM

Programmierern eröffnet GEM neue Perspektiven. Daß die Programmierung von GEM unter C nicht schwierig ist, zeigen wir anhand eines Beispielprogrammes.

Der Atari 520 ST hat ohne Frage eine neue Dimension am Personal Computer-Markt aufgetan. Er ist die erste wirklich preisgünstige Alternative zu den bisherigen Computern mit 68000-CPU. Und er ist auch der bisher einzige 68000-Computer, der GEM unterstützt und dessen Vielseitigkeit in jeder Beziehung ausschöpfen kann. Dieser Vorzug gibt Anlaß, sich grundlegend mit der GEM-Programmierung zu beschäftigen.

»C« von Digital Research war die erste Programmiersprache, die GEM unterstützte. Dadurch schien es naheliegend, »C« auch für die Programmierung unter GEM zu benutzen.

Die wesentlichen Merkmale der GEM-Programmierung bestehen in der bedienerfreundlichen Benutzeroberfläche mit Maus, Fenstertechnik und Drop-Down-Menüs. Schrittweise geben wir hier Einblick in die Programmierung unter GEM, und somit in die hauptsächlichen Unterschiede zwischen »konventioneller«- und GEM-Programmierung. Einige der wichtigsten GEM-Routinen aus der
- Event Library
- Window Library
- Object Library

werden gesondert aufgeführt und so dargestellt, daß jeder mit dem Programmbeispiel experimentieren kann.

Aufbau einer GEM-Anwendung

Eine GEM-Anwendung, oder auch eine Applikation, beginnt in jedem Fall mit der Festlegung globaler Variablen. Sie sind als Übergabeparameter für GEM unbedingt Voraussetzung. Anschließend erfolgt die Anmeldung des Programmes durch die Funktion »appl_init()«. Damit erhält die Anwendung eine Kennziffer zugewiesen, die sie eindeutig identifiziert. Das ist sehr wichtig, da GEM mehrere Applikationen gleichzeitig verwalten kann. Nun belegt man die entsprechenden Variablen vor und öffnet die Bildschirm-Arbeitsstation durch den Funktionsaufruf »v_opnvwk (work_in, &handle, work_out)«. Mit »work_in« setzt man die Parameter der Arbeitsstation. Die Funktion »work_out« zeigt eine Liste von Rückgabewerten, in der besondere Eigenschaften der Arbeitsstation aus GEM zurückgegeben werden könnnen. »handle« kennzeichnet die Arbeitsstation und wird von der Funktion »v_opnvwk (..)« vergeben (diesen Vorgang zeigt Listing 1).

Aufbau von Fenstern

Als nächster Schritt erfolgt der Aufbau von Fenstern. Sämtliche Fensterfunktionen sind in einer Window-Library zusammengefaßt. Die Funktion »wind_create« reserviert zunächst den Speicherbereich, in den die Eigenschaften des Fensters einzutragen sind. So werden die spezifischen Merkmale eines »windows« definiert und es entsteht zum Beispiel ein Schließfeld (Closer), ein Bewegungsfeld (Mover), ein Größenfeld (Sizer) oder ein Rollkasten (Slider). Diese Besonderheiten liegen in der Variablen »kind« fest, wie nebenstehende Tabelle zeigt. Weiterhin sind noch die maximale Größe (maxw, maxh), sowie die Anfangskoordinaten des aufzubauenden Fensters zu vergeben. Die X- und Y-Koordinaten (maxx/maxy) positionieren die linke obere Ecke des Fensters. Wiederum eine Kennziffer benötigt die Funktion »window-handle«, die dann folgendermaßen lautet: »wi_handle=wind_create (kind, maxx, maxy, maxw, maxh)«.

Nun kann man das vorher definierte Fenster am Bildschirm editieren (Listing 2). Mit dem Befehl »wind_open (wi_handle, wx, wy, ww, wh)« richtet man das Fenster nach X- und Y-Koordinate (wx/wy), sowie Breite und Höhe (ww/wh), aus. Es kann noch nachträglich geändert oder sein Zustand nach Lage, Größe und Name etc. abgefragt werden.

Eine Änderung erfolgt über den Befehl »wind_set (wi_handle, setfield, sw1, sw2, sw3, sw4,)«, wobei der »wi_

```
/*************************************************************************/
/* Beispiel: offnen einer Bildschirm-Arbeitsstation in einer Prozedur   */
/*************************************************************************/
open_screen()
{
int    i;
   for(i=0;i<10;work_in[i++]=1);
   work_in[10]=2;
   v_opnvwk(work_in,&handle,work_out);
}
```

Listing 1. Öffnen einer Arbeitsstation

```
/*************************************************************************/
/*  Beispiel:  öffnen eines  Fensters in einer Prozedur                 */
/*************************************************************************/
open_window(xinit,yinit,winit,hinit)
int    xinit,yinit,winit,hinit;
{
   wi_handle=wind_create(CLOSER|NAME|MOVER,xdesk,ydesk,wdesk,hdesk);
   wind_set(wi_handle,WF_NAME," TEST ",0,0);
   wind_open(wi_handle,xinit,yinit,winit,hinit);
}
```

Listing 2. Öffnen eines Fensters

handle« die Kennziffer des Fensters und »setfield« eine Variable für den Teil, der verändert werden soll, bezeichnet. Die Variablen »sw1-4« sind abhängig von »setfield«, je nachdem welche Variation eintreten soll.

Der Zustand des Fensters ist über »wind__get (wi__handle, getfield, gw1, gw2, gw3, gw4)« zu erfahren. Die Variable »get__field« ähnelt in ihrer Bedeutung dem »set__field« bei »window-set«. Diese Variable bestimmt, welche Funktionen des Fensters abgefragt werden können. Die genaue Funktion ist in Tabelle 1 aufgeführt.

Das geöffnete Fenster läßt sich durch »wind__close« (wi__handle) vom Bildschirm entfernen, und jederzeit durch den Befehl »wi__open« mit der entsprechenden handle-Nummer wieder öffnen. Ein Fenster aus dem reservierten Speicherbereich wieder zu löschen, erledigt der Befehl »wind__delete (wi__handle)«. Dieser Vorgang ist dann unwiderruflich, und die handle-Nummer kann GEM wieder neu vergeben.

Weitere Befehle ermöglichen die Bearbeitung des Fensters, wie zum Beispiel »wind__update«, »wind__calc« und »wind__find«.

Einer der bedeutendsten Unterschiede zu herkömmlicher Programmierung besteht im Einsatz des sogenannten »Multi-Event«, die gewisse Zustände registriert und darauf reagiert. Folgende Funktionen werden überwacht:

- ist eine Maustaste gedrückt und welche Position hat der Maus-Cursor?
- wurde eine Taste betätigt?
- steuert man den Maus-Cursor in ein oder aus einem vordefinierten Rechteck?
- ist eine eingestellte Zeit abgelaufen?
- ist ein Menü ausgewählt oder ein Fenster verändert worden?

Der Multi-Event-Funktionsaufruf setzt das Programm in Warteposition auf eine sogenannte »Not-Ready-Liste«. Darin befinden sich bereits weitere angemeldete Programme. Tritt das »erwartete Ereignis« ein, werden alle Programme, die sich im Wartezustand befanden, auf die »Ready-Liste«

Wenn in der Variablen »kind« die entsprechenden Bit gesetzt sind, hat ein Fenster die folgenden Funktionen:

Bit	Name	Funktion
Bit	0:NAME	(Titelbalken)
Bit	1:CLOSER	(Schließfeld)
Bit	2:FULLER	(Vollfeld)
Bit	3:MOVER	(Bewegungsfeld)
Bit	4:INFO	(Informationszeile)
Bit	5:SIZE	(Größenfeld)
Bit	6:UPARROW	(Rollpfeil nach oben)
Bit	7:DNARROW	(Rollpfeil nach unten)
Bit	8:VSLIDE	(vertikaler Rollkasten)
Bit	9:LFARROW	(Rollpfeil nach links)
Bit	10:RTARROW	(Rollpfeil nach rechts)
Bit	11:HSLIDE	(horizontaler Rollkasten)

»wind__get (wi__handle, getfield, gw1, gw2, gw3, gw4)« (Für »getfield« ist die entsprechende Kennziffer einzusetzen).

»wind set (wi__handle, setfield, sw1, sw2, sw3, sw4)«. (Für »setfield« ist die entsprechende Kennziffer einzusetzen).

set__field (Kennziffer)		Bemerkungen
WF__KIND	(1)	In der Variablen sw1 wird ein Bitfeld angegeben, das anzeigt, welche neuen Merkmale das Fenster haben soll. (Siehe »wind__create«).
WF__NAME	(2)	Sw1 und sw2 zeigen auf ein »String«, der den neuen Namen der Titelzeile enthält.
WF__INFO	(3)	Sw1 und sw2 zeigen auf ein »String«, der in der Infozeile dargestellt wird.
WF__NEWDESK	(4)	Zum Zeichnen eines neuen GEM-Desktop. sw1 und sw2= Adresse des Objektbaumes. sw3= Index des Startobjektes.
WF__CURRXYWH	(5)	
WF__HSLIDE	(8)	
WF__VSLIDE	(9)	
WF__TOP	(10)	siehe »get__field«
WF__HSLSIZE	(15)	
WF__VSLIZE	(16)	

get__field (Kennziffer)		Bemerkungen
WF__WORKXYWH	(4)	Gibt die Größe des Arbeitsbereiches an. gw1=x-Koordinate, gw2=y-Koordinate, gw3= Breite, gw4= Höhe
WF__CURRXYWH	(5)	Gibt die Gesamtgröße des Fensters an. gw1-4: s. o.
WF__PREVXYWH	(6)	Gibt die Koordinaten des vorletzten geöffneten Fensters an. gw1-4 : s. o.
WF__FULLXYWH	(7)	Gibt die größtmöglichen Ausmaße des Fensters an (festgelegt durch »wind__create«). gw1-4: s. o.
WF__HSLIDE	(8)	Gibt die relative Position des horizontalen Rollbalkens an (gw1= Zahl zwischen 1 und 1000)
WF__VSLIDE	(9)	Gibt die relative Position des vertikalen Rollbalkens an (gw1= Zahl zwischen 1 und 1000).
WF__TOP	(10)	gw1=window-handle des aktiven Fensters.
WF__FIRSTXYWH	(11)	Gibt die Größe des ersten Fensters, das geöffnet wurde. gw1=x-Koordinate, gw2=y-Koordinate, gw3=Breite, gw4= Höhe.
WF__NEXTXYWH	(12)	Gibt die Größe des Fensters, das als nächstes geöffnet wurde. gw1-4: s. o.
WF__RESVD	(13)	Keine Bedeutung.
WF__HSLIZE	(15)	Gibt die Größe des horizontalen Rollkastens an. gw1= Zahl zwischen 1 und 1000; relative Größe verglichen mit dem Rollbalken.
WF__VSLIZE	(16)	Gibt die Größe des vertikalen Rollkastens an. gw1= Zahl zwischen 1 und 1000; relative Größe verglichen mit dem Rollbalken.

Tabelle 1. Funktionen oder Variablen »kind«, »setfield/getfield«

gesetzt. Aus dieser werden Programme abgearbeitet, bis Sie durch einen weiteren Multi-Event-Aufruf auf die »Not-Ready-Liste« gelangen. Das stellt quasi eine »Multi-Task«-Fähigkeit dar

Neben dem Multi-Event gibt es noch weitere Events, die jeweils nur eine Abfrage gestatten. Das sind zum Beispiel Maus- oder Tastatur-Event. Wegen seiner Fähigkeit, mehrere Ereignisse auf einmal abzufragen, empfiehlt es sich jedoch, generell den Multi-Event zu benutzen.

Der Multi-Event wird in einer Endlos-Schleife ständig aufgerufen, bis bestimmte Bedingungen den Abbruch bewirken. Das geschieht zum Beispiel durch QUIT im Menü oder dem Schließen eines Fensters. Nach jedem Aufruf enthält die Variable »event« den Wert des aufgetretenen Ereignisses.

```
do{
    event=multi_event (Parameter des Event);
    if (event== Ereignis)
        {Programmteil, der den Event verarbeitet}
    if (event== ...)
} while (Abbruchsbedingung);
```

Mit Hilfe der ermittelten Funktionswerte ruft man die entsprechenden Prozeduren auf, wie »Fenster bewegen«, »auf eine Tastatureingabe reagieren« und so weiter. Auf welches Ereignis der Multi-Event reagieren soll, hängt von den Anforderungen an das Programm ab. Der Aufbau des Funktionsaufrufes:

```
int event;
int msgbuff[8],flag,bclick,...;
event=evnt_multi(flag,
bclick,bmask,bstate,
m1_flag,m1x,m1y,m1w,m1h,
m2_flag,m2x,m2y,m2w,m2h,
msgbuff,lo_count,hi_count,
&mx,&my,&m_button,&kstate,
&key_ret,&b_ret);
```

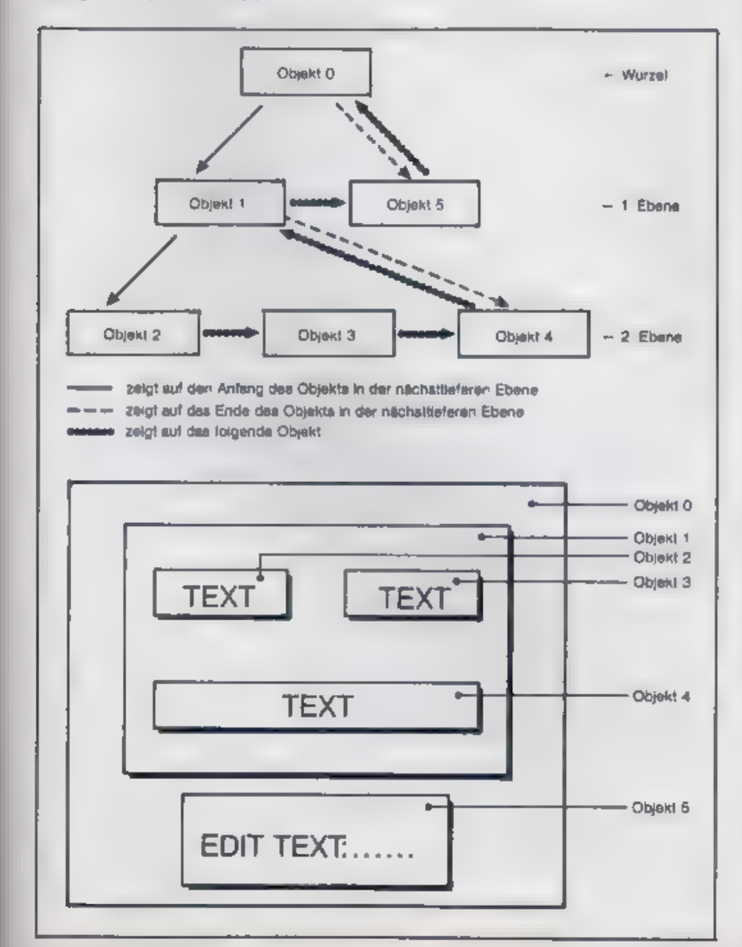


Bild 1. Objektbaum

Die Bitkombinationen in der Variablen »flag« geben an, welche Ereignisse zu erwarten sind. Folgende Funktionen lassen sich mit dem Multi-Event überwachen:

MU_KEYBD (bit0): wartet auf eine Tastatureingabe und gibt den Wert der gedrückten Taste zurück.

MU_BUTTON (bit1): wartet auf die Bestätigung der Maustasten und gibt an, welche Maustaste gedrückt wurde und wo sich der Maus-Cursor befindet.

MU_M1 (bit2): wartet darauf, daß der Maus-Cursor sich in ein vorgegebenes Rechteck bewegt oder es verläßt.

MU_M2 (bit3): wie bei MU_M1

MU_MESAG (bit4): wartet auf eine Mitteilung vom GEM (Auswahl eines Menüeintrages oder die Veränderung eines Fensters). Dabei übernimmt GEM die Handhabung des Menüs:

(»Aufklappen«, »Zuklappen«), der Slider, Mover und Sizer (Bewegen der Umrandungen, bis man die Maustaste losläßt). Die Ausgabe-Variable »msgbuff« ist ein acht Wörter großes Feld (int msgbuff[8]). Das erste Wort bezeichnet die Kennziffer des Ereignisses, das zweite die Applikationskennziffer, das dritte die Länge der Mitteilung. Die weiteren Wörter enthalten die für diesen Aufruf wichtigen Daten.

MU_TIME (bit5): wartet auf den Ablauf einer vom Programmierer gesetzten Zeit.

Objekt-Struktur

GEM bietet zwei Möglichkeiten, um Icons, Dialogboxen und ähnliches zu erzeugen und zu verändern: zum einen mit Hilfe der Resource-Library, und zum anderen mit der Objekt-Library. Prinzipiell besteht der Unterschied zwischen der Resource- und der Objekt-Library nur in der Erzeugungsart der Objekte. Bei der Resource-Library bedient man sich eines Resource-Managers bedienen, der die Objekte direkt auf dem Bildschirm erzeugt und verändert und sie in einem Resource-File speichert.

So lassen sich direkt vom Anwender-Programm aus die Objekte in den Speicher laden. Das bietet den Vorteil, die Objekt-Struktur außerhalb des Programms lagern zu können und sie, ohne in den Quelltext des Programms eingreifen zu müssen, zu ändern. Das ist zum Eindeutschen von Menüs oder Dialog-Boxen notwendig. Die Object-Library bietet diesen Vorteil nicht. Bei ihr muß man die Objekt-Struktur im Quelltext des Programms definieren oder vom Programm selbst erzeugen lassen. Dafür kann man mit ihr Objekt-Typen erzeugen, die nicht standardmäßig vorhanden sind.

Objekte darstellen

Die Objekte selbst sind in einer Baumstruktur hierarchisch geordnet. Sie verfügen über drei »Zeiger«: Der erste zeigt auf das folgende Element in gleicher Ebene. Der zweite und dritte Zeiger weist auf Anfang und Ende der Objekte der nächsttieferen Ebene (Bild 1). Die Zeiger und weitere beschreibende Charakteristika weisen eine bestimmte Struktur auf. Sie enthält Zeiger, die auf Strukturen deuten (Tabelle 2).

Mit Hilfe der Library lassen sich die Objektbäume erzeugen und verändern.

Objektbäume erzeugen

Die Object-Library bietet hierzu drei Routinen:

»Objc_add (tree, parent, child)« fügt in einem Objektbaum »tree« an das Objekt »parent« ein weiteres mit dem Namen »child« an.

»Objc__delete (tree, object)« entfernt aus der Baumstruktur »tree« ein Objekt.

»objc__order (tree, object, new__pos)« bewegt das Objekt in dem Objektbaum »tree« zu einer neuen Position »new__pos«.

Verarbeiten der Objekte

Um die erzeugten Objekte zu verarbeiten (darzustellen, zu finden), stehen verschiedene Funktionen aus der Object-Library zur Verfügung:

»objc__draw (tree, startob, depth, xclip, yclip, wclip, hclip)« zeichnen die Objekte aus dem Objektbaum »tree«. Sie beginnt mit »startob«, wobei die Tiefe »depth« angibt, bis zu welcher Ebene im Objektbaum gezeichnet werden soll. Die X- und Y-Koordinaten, sowie Weite und Höhe (wclip, hclip) beschreiben, welcher Bildschirmbereich für die Darstellung des Objektes freigegeben wird.

»obj__index=objc__find (tree, startob, depth, mx, my)« ermittelt, welches Objekt aus dem Objektbaum »tree« sich unter den angegebenen Bildschirmkoordinaten »mx, my« befindet

»objc__change (tree, object, o, xclip, yclip, wclip, hclip, newstate, redraw)« verändert den Status »newstate« eines Objektes. Durch eine »1« in der Variablen »redraw« wird das Objekt neu gezeichnet. Eine »0« verändert nur den Status im Objektbaum »tree«. Mag das Thema »Objektstruktur« zunächst ein wenig abstrakt erscheinen – gerade die Objekt-Library bietet eine Vielfalt von Merkmalen, die Programmierung erheblich interessanter zu gestalten. Dies gilt in gleichem Maße natürlich für die gesamte C-Programmierung unter GEM. Leider sind die Grundlagen der GEM-Programmierung derart komplex, daß es kaum möglich ist, auf sämtliche Einzelheiten ausführlich einzugehen. Dieser Artikel beschränkt sich daher auf die wichtigsten Grundlagen und soll einen Anreiz geben, mit Hilfe des Beispielprogramms und der dargestellten Fakten zu experimentieren. Auch Compiler-Neulinge sollten ruhig den – zugegeben – nicht ganz einfachen Schritt wagen, sich in diese Art der Programmierung einzuarbeiten. Es lohnt sich! GEM-Programmierung hält eine

OBJEKT-Struktur

Zeiger auf das folgende Objekt: (ob__next).	Zeiger auf den Anfang der Objekte in der nächsttieferen Ebene: (ob__head).
Zeiger auf das Ende der Objekte in der nächsttieferen Ebene: (ob__tail).	Objekttyp: (G__BOX, G__BOXTEXT...) (ob__type).
Objektflags: (ob__flags) (NONE, LASTOB, ...).	Objektstatus: (ob__state) (NORMAL, SELECTED, ...).
Bei den Objekttypen G__BOX, G__IBOX und G__BOXCHAR bestimmt dieses Langwort die Dicke der Umrandung der Box und die Farbe. Bei allen anderen Objekttypen zeigt dieses Wort auf die objektspezifischen Datenstrukturen: (ob__spec)	
X-Position des Objektes:(ob__x)	Y-Position des Objektes:(ob__y)
Weite des Objektes: (ob__w).	Höhe des Objektes:(ob__h)

Die Bildschirmposition des Objektes wird relativ zum Vorgängerobjekt angegeben. Handelt es sich um die »Wurzel«, so gibt sie die Position relativ zur oberen linken Ecke des Bildschirms an.

Strukturdefinition in »C«

```
typedef struct object
 {
         int          ob_next;
         int          ob_head;
         int          ob_tail;
         unsignet int ob_type;
         unsignet int ob_flags;
         unsignet int ob_state;
         long         ob_spec;
         int          ob_x;
         int          ob_y;
         int          ob_w;
         int          ob_h;
} OBJECT;
```

Tedinfo-Struktur:

Zeiger auf den aktuellen Text: (te__ptext).	
Zeiger auf einen Text, der als Maske fungiert: (te__ptmplt).	
Gibt an, welche Zeichen erlaubt sind (te__pvalid).	
Gibt die Schriftgröße an: 3= normal; 5 = klein (te__font).	Hate keine Bedeutung: (te__resvd1).
Gibt an, wie der Text justiert wird:0=links; 1=rechts; 2=mitte (te__just).	Farbe des Textes: (te__color).
Hat keine Bedeutung: (te__resvd2).	Gibt die »Dicke« der Box an, die den Text umrandet.
Die Länge des Strings auf den te__ptext zeigt: (te__txtlen).	Die Länge des Strings auf den »te__ptmplt« zeigt:(te__tmplen).

Es würde zu weit führen, an dieser Stelle ausführlicher zu werden. Nähere Einzelheiten sind der einschlägigen Fachliteratur zu entnehmen.

Strukturdefinition in »C«

```
typedef struct tedinfo
 {
        long        te_ptext;
        long        te_ptmplt;
        long        te_pvalid;
        int         te_font;
        int         te_junk1;
        int         te_just;
        int         te_color;
        int         te_junk2;
        int         te_thickness;
        int         te_txtlen;
        int         te_tmplen;
 } TEDINFO;
```

Tabelle 2. Darstellung der Objektstruktur

Unmenge von Funktionen bereit, die eine neue Dimension in der Programmierung eröffnen.

Dieser Abschnitt beschreibt die Ein- und Ausgabevariablen des Multi-Event. Anhand des Beispielprogrammes »Puzzle« ist ein Multi-Event genau dargestellt.

Aufbau des Funktionsaufrufes.

```
int event;
int msgbuff[8],flag,bclick,...;
event=evnt_multi(flag, bclick,bmask,bstate,
m1_flag,m1x,m1y,m1w,m1h,m2_flag,m2x,m2y,m2w, m2h,
msgbuff,lo_count,hi_count,&mx,&my,&m_button,
&kstate,&key_ret, &b_ret);
```

Die Bitkombinationen in der Variablen »flag« geben an, welche Ereignisse erwartet werden.

Bit 0: MU__KEYBD wartet auf eine Tastatureingabe und gibt den Wert der gedruckten Taste in »key__ret« zurück.

Bit 1: MU__BUTTON kontrolliert die Betätigung der Maustasten. Dazu benötigt es drei Eingabeparameter:
»bclick« – Gibt die Anzahl der erwarteten »Klicks« an
Wert 1 = »einfachklick«
Wert 2 = »doppelklick«
»bmask« – Gibt die benutzten Maustasten an.

Bit 0 =linker Mausknopf
Bit 1 =rechter Mausknopf
Wert 1 =aktiv
Wert 0 =nicht aktiv
»bstate« enthält die Eingabebedingung der Mausknöpfe:

Bit 0 =linker Mausknopf
Bit 1 =rechter Mausknopf
Wert 1 =Mausknopf gedrückt
Wert 0 =Mausknopf nicht gedrückt.

Folgende Parameter sind die Antwort:
»mx,my« – x- und y-Koordinate des Mauscursors
»b__ret« – Anzahl der Mausklicks zwischen 1 und bclick
»m__button« – Betätigte Mausknöpfe:
Bit 0 =linker Mausknopf
Bit 1 =rechter Mausknopf
Wert 1 =Knopf gedrückt
Wert 0 =Knopf nicht gedrückt

»kstate« – Status der Tasten ohne ASCII-Code:
Bit 0 =Shift rechts
Bit 1 =Shift links
Bit 2 =Control
Bit 3 =Alternate
Ist das Bit gesetzt und die Taste gedrückt.

Bit 2: MU__M1 wartet darauf, daß der Mauscursor ein vorgegebenes Rechteck betritt oder verläßt. Dazu werden fünf Eingabeparameter benötigt.

»m1__flag« 0 = wartet darauf, daß man den Mauscursor in das Rechteck steuert
1 = Wartet darauf, daß der Mauscursor das Rechteck verläßt
»m1x,m1y« x- und y-Koordinate der oberen linken Ecke.
»m1w,m1h« Weite und Höhe des Rechtecks.

Zurückgegeben werden die gleichen Parameter wie bei MU__BUTTON, mit Ausnahme von b__ret.

Bit 3: MU__M2 erfüllt dieselbe Funktion wie Bit 2. Dazu werden die gleichen Ein- und Ausgabeparameter wie bei MU__M1 benutzt.

Bit 4: MU__MESAG wartet auf Mitteilungen vom GEM, wie beispielsweise die Auswahl eines Menüeintrages oder die Veränderung eines Fensters. Bei der Verwendung von MU__MESAG sind keine Eingabeparameter notwendig. Der Ausgabeparameter »msgbuff« ist ein Feld mit acht Wörtern (int msgbuff[8]), dessen erstes Wort die Kennziffer des Ereignisses, das zweite Wort die Applikationsziffer enthält. Das dritte Wort enthält die Länge der Mitteilung und die weiteren Wörter, die die für die Ausführung wichtigen Daten beinhalten. Im folgenden sind alle Mitteilungen beschrieben, die bei einem »MU__MESAG-EVENT« eintreten können.

MN__SELECTED: Teilt der Applikation mit, daß vom Benutzer ein Menüeintrag ausgewählt wurde. Dabei übernimmt GEM die Handhabung des Menüs (Aufklappen, Schließen nach Auswahl). Die Applikation hat sich lediglich darum zu kümmern, den ausgewählten Menütitel wieder auf Normalzustand zu setzen (schwarze Schrift auf weißem Hintergrund).
wort 0 =10
wort 3 =Objekt-Index des ausgewählten Menütitels.
wort 4 =Objekt-Index des ausgewählten Menüeintrags

WM__REDRAW: Es ist ein Ereignis eingetreten, das die Wiederherstellung eines Teils der Arbeitsfläche in einem Fenster erfordert.
wort 0 =20
wort 3 =Window Handle (Kenn-Nummer des wiederherzustellenden Fensters)
wort 4 =x-Koordinate dieses Fensters
wort 5 =y-Koordinate dieses Fensters
wort 6 =Weite des Teils
wort 7 =Höhe des Teils

WM__TOPPED: es wurde ein nicht aktives Fenster »angeklickt«.
wort 0 =21
wort 3 =window handle

WM__CLOSED: Ein Closer (Schließer) wurde »angeklickt«.
wort 0 =22
wort 3 =window handle

WM__FULLED: Meldet das »Anklicken« eines »Fuller«. Das Fenster wird auf die volle Größe gebracht, oder, wenn es diese Größe bereits hat, auf die vorherige Größe wieder reduziert.
wort 0 =23
wort 3 =window handle

WM__ARROWED: Es wurde ein Pfeil oder die »Scrollbar« angewählt. Mit dem Pfeil soll die Arbeitsfläche zeilen- oder spaltenweise verschoben werden. »Scrollbar« erledigt dies seitenweise.
wort 0 =24
wort 3 =window handle
wort 4 =0-Seite nach oben
1-Seite nach unten
2-Zeile nach oben
3-Zeile nach unten
4-Seite nach links
5-Seite nach rechts
6-Spalte nach links
7-Spalte nach rechts.

WM__HSLID: Ein horizontaler »Slider« wurde verschoben. GEM kontrolliert dabei die Bewegung des Sliders, bis die Maustaste losgelassen wird und übergibt anschließend die neue Position.
wort 0 =25
wort 3 =window handle
wort 4 =Zahl zwischen 0 und 1000. Dabei bezeichnet 0 die linke und 1000 die rechte Position des Sliders.

WM__VSLID: Der vertikale Slider wurde verschoben.
wort 0 =26
wort 3 =window handle
wort 4 =Zahl zwischen 0 und 1000, 0 ist die oberste und 1000 die unterste Position des Sliders.

WM__SIZED: Die »Size-box« wurde betätigt. GEM überwacht die Bewegung, bis die Maustaste losgelassen wird und übergibt dann die neue Größe.

wort 0 =27
wort 3 =window handle
wort 4 =x-Koordinate des Fensters
wort 5 =y-Koordinate des Fensters
wort 6 =neue Weite des Fensters
wort 7 =neue Höhe des Fensters

WM__MOVED: Fenster wurde verschoben
wort 0 =28
wort 3 =window handle
wort 4 =neue x-Koordinate
wort 5 =neue y-Koordinate
wort 6 =Weite
wort 7 =Höhe

WM__NETOP: Hiermit wird der Applikation mitgeteilt, daß eines ihrer Fenster aktiviert werden soll. Der Vorgang tritt nach dem Schließen eines aktiven Fensters ein.
wort 0 =29
wort 3 =window handle

AC__OPEN: Es wurde ein »Desk Accessoire« aus dem Menü ausgewählt.
wort 0 =30
wort 3 =Objekt-Index des ausgewählten Menüeintrags

AC__CLOSE: zeigt das Schließen eines »Desk-Accessoire« an. Folgende Bedingungen erzeugen einen AC__CLOSE:
- die Applikation ist beendet.
- der Bildschirm soll gelöscht werden.
- Fensterdaten werden reinitialisiert.

Bit5: MU__TIMER wartet auf den Ablauf einer gesetzten Zeit. Der Übergabeparameter enthält in einem Langwort die Zeit in Millisekunden.

lo__count: Der niederwertige Teil eines Langwortes.
hi__count: Der höherwertige Teil eines Langwortes.

Nachdem nun die Struktur des »Multi-Event« bekannt ist, zeigen wir mit dem Programm »Puzzle«, welche Rolle der »Multi Event« in einer GEM-Applikation spielt und wie man ihn einsetzt (Listing 3).

Das Programmbeispiel ist ein Zahlenpuzzle, das vom Desktop aufgerufen und in einem Fenster dargestellt wird. Die Zahlen (Objekte) lassen sich durch »Anklicken« verschieben, und – mit etwas Geschick – in die richtige Reihenfolge bringen (1 oben links, bis 15; unterste rechte Ecke bleibt frei). In diesem Beispiel sind die wichtigsten Teile eines GEM-Programms aufgeführt.

In diesem Programm werden jedoch auch Funktionen benutzt, die im Artikel nicht beschrieben sind. Es würde den Rahmen sprengen, sämtliche Funktionen aufzuführen. Alle im Artikel beschriebenen Funktionen lassen sich jedoch im Programm wiederfinden, so daß der Leser das Beispielprogramm nachvollziehen kann. Der wesentliche Unterschied zum Artikel besteht darin, daß das Programm vom »Desk-Info« aufgerufen wird und sich so der »Event« in einer endlosen Schleife befindet. (Th. Mürke/hb)

```
/*************************************************************************/
/*            PUZZLE                                                     */
/*                                                                       */
/*************************************************************************/
                                        /*******************************/
extern long   xbios();                  /* OSBIND.O                    */
                                        /*                             */
#define Dosound(a)    xbios(32,a)       /* erzeugt einen Ton           */
#define Random()    xbios(17)           /* erzeugt eine Zufallszahl    */
                                        /*******************************/

#define NO_WINDOW (-1)
#define TRUE       1
#define FALSE      0
#define MASK       0xF

#define MU_BUTTON  0x0002
#define MU_MESAG   0x0010

#define WM_REDRAW  20
#define WM_TOPPED  21
#define WM_CLOSED  22
#define WM_MOVED   28
#define WM_NEWTOP  29
#define AC_OPEN    40
#define AC_CLOSE   41

#define NAME       0x0001
#define CLOSER     0x0002
#define MOVER      0x0008

#define WF_NAME         2
#define WF_WORKXYWH     4
#define WF_CURRXYWH     5
#define WF_TOP          10
#define WF_FIRSTXYWH    11
#define WF_NEXTXYWH     12
```

Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt

```
#define ARROW       0
#define M_OFF     256
#define M_ON      257

#define G_BOX      20
#define G_BOXTEXT  22

#define NONE        0x0
#define LASTOB      0x20

#define NORMAL      0x0
#define SELECTED    0x1

typedef struct object
{                                           /*********************************/
   int       ob_next;                       /* -> object's next sibling      */
   int       ob_head;                       /* -> head of object's children  */
   int       ob_tail;                       /* -> tail of object's children  */
   unsigned int    ob_type;                 /* type of object- BOX, CHAR,...*/
   unsigned int    ob_flags;                /* flags                         */
   unsigned int    ob_state;                /* state- SELECTED, OPEN, ...    */
   long       ob_spec;                      /* "out"- -> anything else       */
   int       ob_x;                          /* upper left corner of object   */
   int       ob_y;                          /* upper left corner of object   */
   int       ob_width;                      /* width of obj                  */
   int       ob_height;                     /* height of obj                 */
} OBJECT;                                   /*********************************/

typedef struct grect                        /*********************************/
{                                           /*                               */
   int   g_x;                               /*  für redraw_windows();        */
   int   g_y;                               /*                               */
   int   g_w;                               /*                               */
   int   g_h;                               /*                               */
} GRECT;                                    /*********************************/

typedef struct text_edinfo
{                                           /*********************************/
   long       te_ptext;                     /* ptr to text (must be 1st)     */
   long       te_ptmplt;                    /* ptr to template               */
   long       te_pvalid;                    /* ptr to validation chrs.       */
   int       te_font;                       /* font                          */
   int       te_junk1;                      /* junk word                     */
   int       te_just;                       /* justification- left,right... */
   int       te_color;                      /* color information word        */
   int       te_junk2;                      /* junk word                     */
   int       te_thickness;                  /* border thickness              */
   int       te_txtlen;                     /* length of text string         */
   int       te_tmplen;                     /* length of template string     */
} TEDINFO;                                  /*********************************/

/***************************************************************************/
/*    GLOBALE VARIABLEN                                                    */
/***************************************************************************/

                                            /*****************************/
extern int  gl_apid;                        /* Idetifikation des Prg's   */
                                            /*****************************/

                                            /*****************************/
int   xdesk,ydesk,wdesk,hdesk;              /* GRöSSE DES DESKTOP        */
int   xwork,ywork,wwork,hwork;              /* GRöSSE DER ARBEITSFLÄCHE  */
                                            /*****************************/
```

Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt. (Fortsetzung)

```
int   contrl [12];                          /******************************/
int   intin [128];                          /*       GEM - AES/VDI        */
int   ptsin [128];                          /*         VARIABLEN          */
int   intout[128];                          /*                            */
int   ptsout[128];                          /******************************/

                                            /******************************/
int   handle;                               /*  VDI-HANDLE                */
                                            /******************************/

                                            /******************************/
int   wi_handle;                            /* HANDLER FÜR OBJEKTFENSTER */
                                            /******************************/

int   work_in [11];                         /******************************/
int   work_out[57];                         /* INPUT/OUTPUT DES GSX       */
                                            /******************************/

                                            /******************************/
int   menu_id;                              /*  Menü Identifikator        */
int   top_window;                           /******************************/

                                            /******************************/
int   hidden;                               /* Cursor Status              */
                                            /******************************/

                                            /******************************/
int   top_window;                           /* Identifikator des aktiven */
                                            /* Fensters                   */
                                            /******************************/

                                            /******************************/
int   buf[15];                              /* Zufällige Reihenfolge      */
                                            /* der Tedinfo-Zeiger         */
                                            /******************************/

                                            /******************************/
char  null[]="";                            /* LEERE ZEICHENKETTE         */
                                            /*                            */
char  string[][]={"1","2","3","4","5",      /* STRINGS FÜR TEDINFO        */
                  "6","7","8","9","10",     /*                            */
                  "11","12","13","14",      /*                            */
                  "15"};                    /******************************/

/*************************************************************************/
/* Initialisiere tedinfo                                                 */
/*************************************************************************/
TEDINFO tedinfo[] = {
string[0],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[2],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[4],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[6],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[8],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[10],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[12],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[14],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[16],null,null, 3, 6, 2, 0x1180, 0x0, -1, 2,0,
string[18],null,null, 3, 6, 2, 0x1180, 0x0, -1, 3,0,
string[21],null,null, 3, 6, 2, 0x1180, 0x0, -1, 3,0,
string[24],null,null, 3, 6, 2, 0x1180, 0x0, -1, 3,0,
string[27],null,null, 3, 6, 2, 0x1180, 0x0, -1, 3,0,
string[30],null,null, 3, 6, 2, 0x1180, 0x0, -1, 3,0,
string[33],null,null, 3, 6, 2, 0x1180, 0x0, -1, 3,0};
```

Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt (Fortsetzung)

```
/*************************************************************************/
/* Initialisiere object                                                  */
/*************************************************************************/
OBJECT object[] = {
-1, 1, 16, G_BOX, NONE, NORMAL, 0x1121L , 100,100, 87,87,
2, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 2,2, 20,20,
3, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 23,2,20,20,
4, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 44,2,20,20,
5, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 65,2,20,20,
6, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 2,23,20,20,
7, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 23,23,20,20,
8, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 44,23,20,20,
9, -1, -1, G_BOXTEXT, NONE, NORMAL, null, 65,23,20,20,
10, -1, -1, G_BOXTEXT, NONE, NORMAL,null, 2,44,20,20,
11, -1, -1, G_BOXTEXT, NONE, NORMAL,null, 23,44,20,20,
12, -1, -1, G_BOXTEXT, NONE, NORMAL,null, 44,44,20,20,
13, -1, -1, G_BOXTEXT, NONE, NORMAL,null, 65,44,20,20,
14, -1, -1, G_BOXTEXT, NONE, NORMAL,null, 2,65,20,20,
15, -1, -1, G_BOXTEXT, NONE, NORMAL,null, 23,65,20,20,
16, -1, -1, G_BOXTEXT, NONE, NORMAL,null, 44,65,20,20,
 0, -1, -1, G_BOX,   LASTOB, NORMAL,0x1121L,65,65,20,20};

/*************************************************************************/
/* Setzt die x- und y-Koordinaten der Objecte auf den Startwert          */
/*************************************************************************/
resetxy()
{
int   c,z,p;
   p=0;
   for(c=2;c<=65;c+=21){
      for(z=2;z<=65;z+=21){
         object[++p].ob_x=z;
         object[p].ob_y=c;
      }
   }
}

/*************************************************************************/
/* Setzt die TEDINFO-Zeiger                                              */
/* Z.B. object[2] mit tedinfo[3] ==>                                     */
/*      das dritte Objekt stellt die Zahl 4 auf                          */
/*      der Position x=23,y=2 dar.                                       */
/*************************************************************************/
init_puzzle()
{
int   c,z,in,p;

   objc_change(object,0,0,xwork,ywork,wwork,hwork,NORMAL,1);
   for(c=0;c<15;c++){
      do{
        in=0;
        z=Random() & MASK;
        for(p=0;p<c;p++)if((z==buf[p])||(z>14))in=1;
      }while(in);
      buf[c]=z;
   }
   for(c=0;c<15;object[c+1].ob_spec= &tedinfo[buf[c++]]);
   obj_redraw(0);
}
/*************************************************************************/
/* uberprufe ob sich die Objekte mit den entsprechenden TEDINFO's        */
/* an der richtigen Stelle befinden.                                     */
/* Z.B. OBJECT[7] mit TEDINFO[2] (stellt die 3 dar) ==>                  */
/*      dann mussen object[7].ob_x=44 und object[7].ob_y=2 sein,         */
/*************************************************************************/
```

Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt (Fortsetzung)

```
test_puzzle()
{
int    x,y,i,ready;
   i=0;ready=1;
   do{
      x=buf[i];  y=0;
      if(x > 3){x-=4;y++;}
      if(x > 3){x-=4;y++;}
      if(x > 3){x-=4;y++;}
      if(object[++i].ob_x != x*21+2)ready=0;
      if(object[i].ob_y != y*21+2)ready=0;
   }while((ready)&&(i<15));
   return(ready);
}
/*************************************************************************/
/* Bringe das ganze Puzzel zum blinken (ROOT-OBJECT)                    */
/*************************************************************************/
flash()
{
int    i,t;
static char sound[]={0,200,1,0,7,254,11,255,12,110,13,9,8,16,255,0};
   Dosound(sound);
   for(t=0;t<30;t++){
      for(i=0;i<10000;i++);
      if(t%2) objc_change(object,0,0,xwork,ywork,wwork,hwork,SELECTED,1);
         else objc_change(object,0,0,xwork,ywork,wwork,hwork,NORMAL,1);

   }
}
/*************************************************************************/
/* Prüfe ob sich das Objekt in der Horizontalen verschieben laßt.       */
/* Wenn ja,korregiere x- und y-Koordinaten des Objekt's                 */
/*************************************************************************/
check_hor(x,y,n)
int    x,y,n;
{
int    sel,b,t,c,ret;
   ret=0;
   for(c=x+(n*4);c!=x;sel=find(c-=n,y))if(sel==16){
                                    b=find(c-n,y);t=object[16].ob_x;
                                    object[16].ob_x=object[b].ob_x;
                                    object[b].ob_x=t;
                                    obj_redraw(16);obj_redraw(b);
                                    ret=1;
                                 }
   return(ret);
}
/*************************************************************************/
/* Prüfe ob sich das Objekt in der Vertikalen verschieben läßt.         */
/* Wenn ja,korregiere x- und y-Koordinaten des Objekt's                 */
/*************************************************************************/
check_ver(x,y,n)
int    x,y,n;
{
int    sel,b,t,c,ret;
   ret=0;
   for(c=y+(n*4);c!=y;sel=find(x,c-=n))if(sel==16){
                                    b=find(x,c-n);t=object[16].ob_y;
                                    object[16].ob_y=object[b].ob_y;
                                    object[b].ob_y=t;
                                    obj_redraw(16);obj_redraw(b);
                                    ret=1;
                                 }
   return(ret);
}
```

Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt (Fortsetzung)

```
/***********************************************************************/
/* Finde das Objekt unter dem Cursor  (-1 = kein Objekt)              */
/***********************************************************************/
find(x,y)
int   x,y;
{
int   sel;
   sel=objc_find(object,0,1,x,y);
   return(sel);
}
/***********************************************************************/
/*  Zeichne Objekt                                                     */
/***********************************************************************/
obj_redraw(sel)
int   sel;
{
   objc_draw(object,sel,1,xwork,ywork,wwork,hwork);
}
/***********************************************************************/
/* Prüfe ob sich das angewählte Objekt verschieben läßt.              */
/***********************************************************************/
handle_puzzle(x,y)
int   x,y;
{
int   sel;
   if((sel=find(x,y))!= -1)
      if((sel!=16)&&(sel!=0))
       if(!check_hor(x,y,21))
        if(!check_hor(x,y,-21))
         if(!check_ver(x,y,21))
          check_ver(x,y,-21);
   if(test_puzzle())flash();
}
/***********************************************************************/
/* Cursor aus                                                          */
/***********************************************************************/
hide_mouse()
{
   if(!hidden){
      graf_mouse(M_OFF,0x0L);
      hidden=TRUE;
   }
}
/***********************************************************************/
/* Cursor an                                                           */
/***********************************************************************/
show_mouse()
{
   if(hidden){
      graf_mouse(M_ON,0x0L);
      hidden=FALSE;
   }
}
/***********************************************************************/
/*    öffne die Bildschirm-Arbeitsstation                              */
/***********************************************************************/
open_screen()
{
int   i;
   for(i=0;i<10;work_in[i++]=1);
   work_in[10]=2;
   handle=graf_handle(&wchar,&hchar,&wbox,&hbox);
   wind_get(0,WF_WORKXYWH,&xdesk,&ydesk,&wdesk,&hdesk);
   v_opnvwk(work_in,&handle,work_out);
   graf_mouse(ARROW,0x0L);
```

Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt (Fortsetzung)

```
}
/**********************************************************************/
/*    öffne das Puzzel-Fenster                                        */
/**********************************************************************/
open_window(xinit,yinit,winit,hinit)
int   xinit,yinit,winit,hinit;
{
   wi_handle=wind_create(CLOSER|NAME|MOVER,xdesk,ydesk,wdesk,hdesk);
   graf_growbox(10,10,20,20,xinit,yinit,winit,hinit);
   wind_set(wi_handle,WF_NAME," PUZZLE ",0,0);
   wind_open(wi_handle,xinit,yinit,winit,hinit);
   wind_get(wi_handle,WF_WORKXYWH,&xwork,&ywork,&wwork,&hwork);
   object[0].ob_x=xwork;
   object[0].ob_y=ywork;
   resetxy();
   init_puzzle();
}
/**********************************************************************/
/* Zeichne alle offenen Fenster neu.                                  */
/**********************************************************************/
redraw_windows(x,y,w,h)
int   x,y,w,h;
{
GRECT t1,t2;
   hide_mouse();
   wind_update(TRUE);
   t2.g_x=x;   t2.g_y=y;
   t2.g_w=w;   t2.g_h=h;
   wind_get(wi_handle,WF_FIRSTXYWH,&t1.g_x,&t1.g_y,&t1.g_w,&t1.g_h);
   while(t1.g_w && t1.g_h){
      if(rc_intersect(&t2,&t1)){
         objc_draw(object,0,1,t1.g_x,t1.g_y,t1.g_w,t1.g_h);
      }
      wind_get(wi_handle,WF_NEXTXYWH,&t1.g_x,&t1.g_y,&t1.g_w,&t1.g_h);
   }
   wind_update(FALSE);
   show_mouse();
}
/**********************************************************************/
/*  multi                                                             */
/**********************************************************************/
multi()
{
int   event,ret,msgbuff[8],mx,my;

   do{
      event=evnt_multi (MU_BUTTON|MU_MESAG,
                        1,1,1,
                        0,0,0,0,0,
                        0,0,0,0,0,
                        msgbuff,0,0,&mx,&my,&ret,&ret,&ret,&ret);

      wind_update(TRUE);
      wind_get(wi_handle,WF_TOP,&top_window,&ret,&ret,&ret);
      if(event & MU_MESAG)
         switch(msgbuff[0]){

         case AC_OPEN:
            if(msgbuff[4] == menu_id){
               if(wi_handle == NO_WINDOW){
               open_screen();
               open_window(270,140,89,107);
            }
         else
```

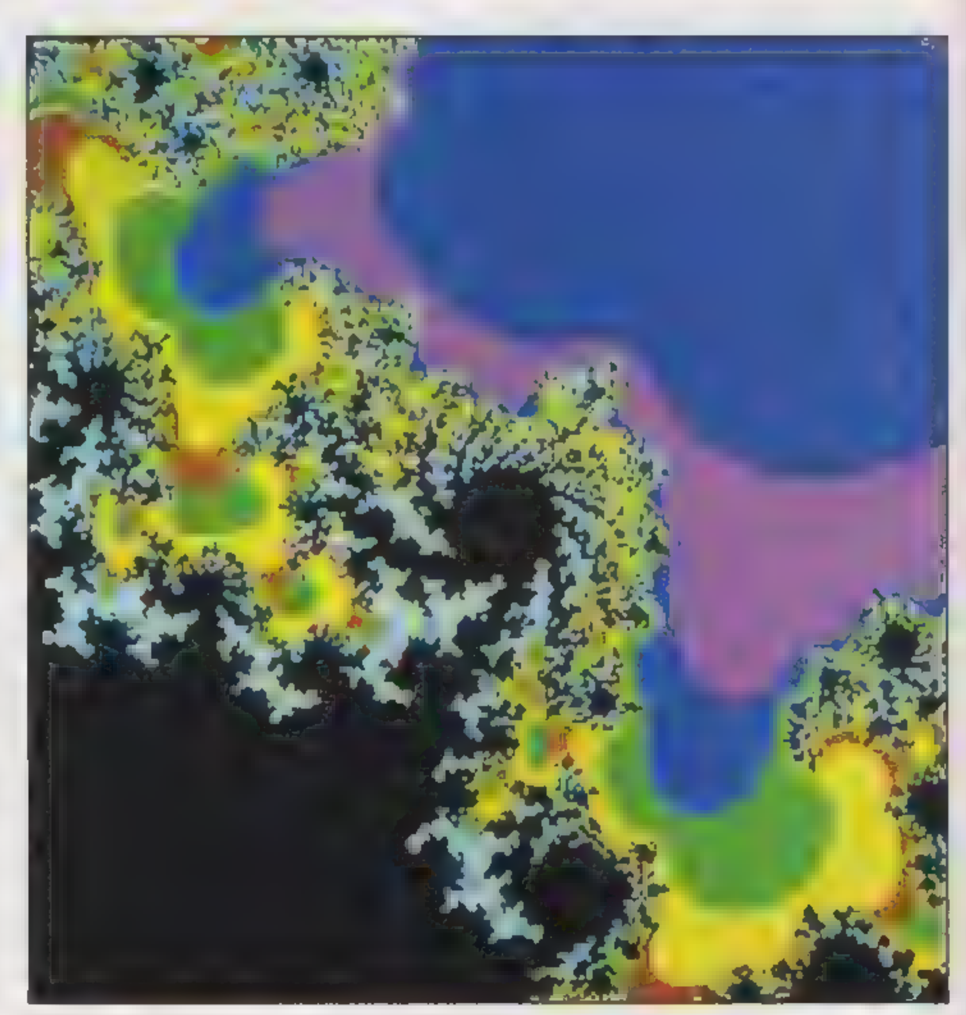
Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt (Fortsetzung)

```
            wind_set(wi_handle,WF_TOP,0,0,0,0);
         }
         break;

      case WM_REDRAW:
         if(msgbuff[3] == wi_handle)
            redraw_window(msgbuff[4],msgbuff[5],msgbuff[6],msgbuff[7]);
         break;

      case WM_NEWTOP:
      case WM_TOPPED:
         if(msgbuff[3] == wi_handle)
            wind_set(wi_handle,WF_TOP,0,0,0,0);
         break;

      case WM_MOVED:
         if(msgbuff[3] == wi_handle){
            wind_set(wi_handle,WF_CURRXYWH,msgbuff[4],msgbuff[5],
                        msgbuff[6],msgbuff[7]);
            wind_get(wi_handle,WF_WORKXYWH,&xwork,&ywork,&wwork,&hwork);
            object[0].ob_x=xwork;    object[0].ob_y=ywork;
         }
         break;

      case AC_CLOSE:
         if((msgbuff[3] == menu_id) && (wi_handle != NO_WINDOW)){
            v_clsvwk(handle);
            wi_handle=NO_WINDOW;
         }
         break;

      case WM_CLOSED:
         if(msgbuff[3] == wi_handle){
            wind_close(wi_handle);
            graf_shrinkbox(20,20,20,20,xwork,ywork,wwork,hwork);
            wind_delete(wi_handle);
            v_clsvwk(handle);
            wi_handle=NO_WINDOW;
         }
         break;

    }/* switch */

   wind_update(FALSE);

   if(event & MU_BUTTON)
      if(object[0].ob_state != SELECTED)
         handle_puzzle(mx,my);
      else { resetxy(); init_puzzle();}

   }while(TRUE);
}
/***************************************************************************/
/* Initialisiere das Programm und Trage das Wort 'Puzzle'                 */
/* ins Desktop Menü ein.                                                   */
/***************************************************************************/
   main()
   {
      appl_init();
      menu_id=menu_register(gl_apid,"  Puzzle ");
      wi_handle=NO_WINDOW;
      hidden=FALSE;

      multi();

   }
```

Listing 3. »Puzzle« (»Puzzle.Doc«) zeigt, welche Rolle der »Multi-Event« in einer GEM-Applikation spielt (Schluß)

Äpflein, Äpflein

Seit in der Oktobernummer 1985 von Spektrum der Wissenschaft eine genaue Beschreibung des Algorithmus erschien, mit dem man auf einem Computer die sogenannte »Apfelmännchen«-Grafik erzeugen kann, wurde die Welt von einer Fülle von Apfelmännchen-Programmen überschwemmt. Hier auch eines.



Blick in das Universum einer Mandelbrot-Menge

Wir haben in unser Sonderheft ein solches Programm aufgenommen, weil es relativ kurz und einfach und dabei in zweierlei Hinsicht typisch für die 68000er-Computer ist: Es verlangt einen hohen Rechenaufwand und eine hochauflösende Bildschirmgrafik. Manche Programme rechnen ganze Nächte, unseres braucht ein bis zwei Stunden und kann die Bilder auch auf Diskette speichern und in wenigen Minuten wieder anzeigen.

Die Farbbilder, die Sie hier sehen, wurden auf einem Stride unter UCSD-Pascal mit einer 8-Farbengrafik mit einer Auflösung von 256 x 256 Punkten erzeugt. Unser Programm liefert auf dem Atari ST Schwarzweißbilder mit 400 x 400 Punkten, die auf einem Epson FX-80 ausgedruckt werden können

Das Programm für den Atari ist in TDI-Modula geschrieben, das sehr schnellen Maschinencode compiliert und vollen Zugriff auf alle Grafikmöglichkeiten von GEM bietet. Das Programm (Listing 1) zeigt nur die allereinfachste Form, Grafik zu programmieren.

Über die Theorie der komplexen Zahlen und die »Mandelbrot-Menge« erfahren Sie in einem eigenen Beitrag Näheres, ich beschränke mich hier auf eine kurze Besprechung des Programms. Sie können das Programm auf Diskette zusammen mit anderen Programmen für den Atari (RAM-Disk, Bagger, Superfilter) von der Schwesterzeitschrift Computer persönlich erhalten. Es gibt zwei Versionen, die eine mit Farbdarstellung 200 x 200 Punkte (File »Apfel.PRG«), die andere Schwarzweiß mit 400 mal 400 Punkten (File »Mandel.PRG«).

Wenn Sie ein wenig Pascal kennen, haben Sie sicher keine Schwierigkeiten, das Programm zu verstehen.

In den Zeilen 9 bis 24 finden Sie eine Reihe von »Importen« (beachten Sie den Artikel: »Pascal + C = Modula-2« in dieser Ausgabe). Es handelt sich dabei um die Anforderung von verschiedenen Bibliotheken für Ein-/Ausgabe. Aus Terminal beziehen wir Prozeduren zum Lesen und Schreiben von einzelnen Zeichen, aus TextIO zum Lesen und Schreiben von Zahlen (INTEGER, CARDINAL). IOUTIL ist ein eigenes Modul (das Sie auch auf der Diskette finden), STREAMS dient dem Lesen und Schreiben von Datenfiles (damit speichert und liest das Programm die Bilder). Schließlich werden aus Mathlib0 die Quadratwurzelfunktion und aus String die Längenfunktion für Zeichenketten benötigt. Speziell für die Grafikausgabe brauchen wir einige Prozeduren aus den GEM-Bibliotheken. Da ist einmal die Funktion GrafHandle aus AESGraphics (Application Environment Support), die uns den »Schlüssel« zum Bildschirm liefert (er muß beim Aufruf der übrigen Grafikprozeduren eingesetzt werden). PolyLine zeichnet Linien – wir verwenden diese Prozedur, weil es eine zum Setzen von einzelnen Punkten nicht gibt. ClearWorkstation löscht grafisch den Bildschirm; SetLineColor dient zum Einstellen der Farbe der Linien, die von PolyLine geschrieben werden. VDI bedeutet Virtual Device Interface.

Das Programm muß an einigen Stellen geändert werden, um es auf die unterschiedliche Auflösung des Farb- oder Schwarzweißbildschirms einzustellen. In erster Linie bestimmen das die Konstanten in den Zeilen 37 bis 45, weitere Abweichungen gibt es in den Prozeduren SetColor (Zeile 102) und Farbe (Zeile 130). Um in SetColor die Struktur der Umgebung des Apfelmännches (den gebänderten Halo) sichtbar zu machen, wird zwischen je zwei Iterationsstufen zwischen Schwarz und weiß umgeschaltet – das geschieht ganz einfach durch die Anweisung color MOD 2. In Zeile 137 wird bei der Schwarzweißversion die Anzahl der Iterationen direkt an SetColor ausgegeben, bei der Farbversion werden die Iterationswerte über mehrere CASE-Anweisungen in eine Farbtabelle übersetzt. Nach dieser Farbtabelle wurden die Farbbilder gewonnen, man kann damit natürlich nach eigenem Geschmack experimentieren.

auf dem Schirm...

Das Hauptprogramm beginnt in Zeile 171, es holt sich zunächst den »Schlüssel« (eine Kennziffer) und schreibt dann ein kleines Menü an. Zuvor wird der Cursor eingeschaltet und der Bildschirm gelöscht. Normalerweise ist unter GEM der Schreibcursor nicht sichtbar, da man mit der Maus arbeitet, die einen eigenen Cursor hat (je nach Situation als Pfeil, Motte oder ähnlich dargestellt). Nach dem Programmstart erscheint ein Menü, aus dem durch einfachen Tastendruck folgende Arbeitsweisen aufzurufen sind

G(enerieren (Aufruf, indem man einfach die Taste »G« drückt): Erzeugen eines neuen Bildes. Das Programm fragt zunächst nach dem Namen einer Bilddatei (Achtung: Wird in der SW-Version 160 000 Bytes groß!). Drückt man nur RETURN, dann wird keine Datei erzeugt. Anschließend werden die Parameter eingegeben. Um das ganze Apfelmännchen zu sehen, gibt man ein: X: -2.0, Y: -1.25, Seite: 2.5. Die Erzeugung eines Bildes kann bis zu mehreren Stunden dauern (abhängig von der maximalen Iteration).

D(atei: Wiederanzeige eines Bildes, das bereits auf Diskette gespeichert wurde – dauert nur wenige Minuten. Laufwerk (A: oder B:) und Dateiname angeben.

Z(aehler einstellen: Bestimmt den Grenzwert der Iteration, ab welcher das »schwarze Loch« definiert wird. Die Anzahl der Iterationen steigt am Rande des schwarzen Lochs steil an, sie verlängert die Rechendauer stark, verfeinert aber – bei Ausschnittvergrößerungen – die sichtbaren Details.

E(nde: Abbruch des Programms, Rückkehr ins TOS.

Der eigentliche Algorithmus nach der Beschreibung in Spektrum der Wissenschaft befindet sich in den Zeilen 210 bis 245. In den Zeilen 218 bis 221 befindet sich ein kleiner Trick: Da der hier abgefragte Bereich sich mit Sicherheit im »schwarzen Loch« befindet, wird er kurzerhand übersprungen. Das beschleunigt die Generierung des Bildes. In Zeile 236 wird ein Bildpunkt geschrieben, der Prozedur **Farbe** wird einfach der Wert des Iterationszählers übergeben. (Sie können die Anzahl der Farben auf 16 erhöhen.)

Wenn eine Datei eröffnet wurde, so ist die Variable Protokoll wahr. Dann wird der Wert des Iterationszählers als Byte auf Diskette geschrieben. Aus diesem Grunde können Dateien nur mit maximalen Iterationswerten von 254 (der Zähler hat im schwarzen Loch dann den Wert 255) gespeichert werden. Vorteilhaft ist, daß man – nach Änderung der Farbübersetzung in der Prozedur Farbe – ein in Stunden erzeugtes Bild mit verschiedener Farbdarstellung anzeigen kann.

In den Zeilen 262 bis 268 befinden sich dieselben geschachtelten FOR-Schleifen für Zeilen (m) und Spalten (n) wie in den Zeilen 212 bis 245, nur wurde hier der Iterationsalgorithmus durch das Lesen der Zählerwerte aus der Datei ersetzt.

Ganz am Schluß findet man zwei unscheinbare, aber wichtige Details: Der Cursor wird wieder ausgeschaltet (Zeile 286), in der darauffolgenden Zeile wird die Farbe wieder auf Schwarz gestellt – sonst wird nämlich das Erscheinungsbild von GEM gestört.(le)

Vom Umriß der Name der fraktalen Figur: Apfelmännchen.

```
 1: MODULE Apfelmaennchen; (* FARBVERSION *)
 2: (* Uebertragen aus UCSD-Pascal *)
 3: (* Idee: Spektrum der Wissenschaft Oktober 1985,
 4:    Columne "Computer-Kurzweil" von A. K. Dewdney *)
 5: (* Realisation: Johannes Leckebusch (Pascal, Modula) *)
 6: (* Uebertragen aus Borland-Modula in TDI-Modula auf Atari *)
 7: (* Stand: 16. 01. 86 *)
 8:
 9: FROM Terminal IMPORT Read, Write;
10: FROM AESGraphics    IMPORT GrafHandle;
11: FROM VDIOutputs     IMPORT PolyLine;
12: FROM VDIControls    IMPORT ClearWorkstation;
13: FROM VDIAttribs IMPORT SetLineColour;
14: (*
15: PROCEDURE SetLineColour ( Handle, ColourIndex : INTEGER ) : INTEGER;
16: (* sets colour index for subsequent polyline operations *)
17: *)
18:
19: FROM IOUTIL IMPORT ReadString, LoescheBild, CursorEin, CursorAus;
20: FROM String IMPORT Length;
21: FROM MathLib0 IMPORT sqrt;
22: FROM Streams IMPORT
23:   StreamKinds (* READ, READWRITE *), Stream, OpenStream, CloseStream,
24:   Read8Bit, Write8Bit, EOS;
25:
26: (*
27: PROCEDURE OpenStream(VAR s: Stream; VAR name: ARRAY OF CHAR;
28:                          kind: StreamKinds;
29:                      VAR reply : INTEGER);
30: PROCEDURE CloseStream(VAR s: Stream; VAR reply: INTEGER);
31: *)
32:
33: FROM TextIO IMPORT ReadReal, ReadInt, ReadCard, WriteInt, WriteCard,
34:                    WriteString, WriteLn;
35:
36:
```

Listing des vollständigen Apfelmännchen-Programms in TDI-Modula. Alle Module sind Bestandteil des Programmierpaketes, mit Ausnahme von »IOUTIL«. Dieses Modul ist auf unserer Service-Diskette.

```
37: CONST   (*
38:         cSpalten =      399;    (* Spalten und Zeilen der *)
39:         cZeilen =       399;    (* Pixel-Matrix schwarzweiss *)
40:         cOffset =       120;    (* quadratische Matrix einruecken *)
41:         *)
42:
43:         cSpalten =      199;    (* Farbe *)
44:         cZeilen =       199;    (* Farbe *)
45:         cOffset =        60;    (* Farbe *)
46:
47:
48:         (* Das Bild wird nicht im Programm gespeichert, sondern gleich
49:            Pixel fuer Pixel auf den Bildschirm bzw. eine Datei
50:            geschrieben *)
51:
52:         cMaxZaehler =   100;    (* maximale Iteration Standardwert *)
53:
54:         weiss =         0;      (* Definition der GEM-Farben *)
55:         schwarz =       1;
56:         rot =           2;
57:         gruen =         3;
58:         blau =          4;
59:         cyan =          5;
60:         gelb =          6;
61:         violett =       7;
62:
63: TYPE    Komplex =       RECORD r, i:    REAL;
64:                         END (* Komplex *);
65:
66:         (* BYTE =          INTEGER [0..255]; *)
67:                         (* Das kennt er wieder nicht... *)
68:
69: VAR     m, mz           CARDINAL;
70:         Zaehler:        CARDINAL (* BYTE *);
71:         MaxZaehler:     CARDINAL (* BYTE *);
72:         c, k, z:        Komplex;
73:
74:         Ecke:           Komplex;
75:         Seite:          REAL;
76:         Spalt:          REAL;
77:         Befehl:         CHAR;
78:
79:         alteFarbe:      (* tPenColor *) CARDINAL;
80:
81:         Datei:          Stream;
82:         Dateiname:      ARRAY [0..30] OF CHAR;
83:         Protokoll:      BOOLEAN;
84:
85: VAR Points:   ARRAY [0..3] OF INTEGER;
86:     handle:   INTEGER;
87:     x, y:     CARDINAL;
[illegible]     dummy:    INTEGER;
89:     ch:       CHAR;
90:
91:   PROCEDURE Sqr (r: REAL): REAL;
92:   BEGIN
93:     RETURN r * r;
94:   END Sqr;
95:
96:   PROCEDURE Betrag (k: Komplex): REAL;
97:   (* Groesse einer komplexen Zahl bestimmen *)
98:   BEGIN
99:     RETURN sqrt (Sqr (k.r) + Sqr (k.i));
100:  END Betrag;
101:
102:  PROCEDURE SetColor (color: CARDINAL);
103:  VAR ok:       INTEGER;
104:    (*
105:    PROCEDURE SetLineColour (Handle, ColourIndex: INTEGER ): INTEGER;
106:    (* sets colour index for subsequent polyline operations *)
107:    *)
108:
109:  BEGIN
110:    (*
111:    ok := SetLineColour (handle, color MOD 2); (* SW *)
112:    *)
113:
114:    ok := SetLineColour (handle, color); (* Farbe *)
115:
116:  END SetColor;
117:
118:  PROCEDURE MoveTo (x, y: CARDINAL);
```

Listing des vollständigen Apfelmännchen-Programms (Fortsetzung)

```
119:  BEGIN
120:    Points [0] := x; Points [1] := y;
121:  END MoveTo;
122:
123:  PROCEDURE DrawTo (x, y: CARDINAL);
124:  BEGIN
125:    Points [2] := x; Points [3] := y;
126:    PolyLine (handle, 2, Points);
127:    Points [0] := Points [2]; Points [1] := Points [3];
128:  END DrawTo;
129:
130:  PROCEDURE Farbe (f, spalte, zeile: CARDINAL);
131:  (* Diese Prozedur setzt das Pixel (spalte, zeile) *)
132:  BEGIN
133:    IF (f > MaxZaehler) (* OR (f < 2) *) THEN
134:      SetColor (schwarz);
135:    END (* IF im schwarzen Loch *);
136:    (*
137:    SetColor (f);       (* !!! Schwarzweiss !!! *)
138:    *)
139:
140:    (* Bestimmen der Pixelfarbe: *)
141:                        (* !!! Farbe !!! *)
142:    IF f < 33 THEN
143:      CASE f MOD 10 OF
144:        0, 1, 2, 3, 4:  SetColor (blau);|
145:        5, 6, 7, 8, 9:  SetColor (violett);|
146:      END (* CASE *)
147:    ELSIF f < 66 THEN
148:      CASE f MOD 10 OF
149:        0, 1, 2:        SetColor (rot);|
150:        3, 4, 5:        SetColor (gruen);|
151:        6, 7, 8, 9:     SetColor (gelb);|
152:      END (* CASE *)
153:    ELSE
154:      CASE f MOD 10 OF
155:        0, 1, 2:        SetColor (gelb);|
156:        3, 4, 5:        SetColor (cyan);|
157:        6, 7, 8, 9:     SetColor (weiss);|
158:      END (* CASE *);
159:    END (* IF *);
160:    (* Bildpunkt schreiben: *)
161:
162:    IF (spalte = 0) THEN
163:      MoveTo (spalte + cOffset, zeile);
164:    END (* IF spalte = 0 *);
165:    DrawTo (spalte + cOffset, zeile);
166:  END Farbe;
167:
168: VAR    MZ:     CARDINAL;
169:        ok:     INTEGER;
170:
171: BEGIN (* Mandelbrot *)
172:   (* Schluessel fuer Ausgabekanal holen: *)
173:   handle := GrafHandle (dummy, dummy, dummy, dummy);
174:   ClearWorkstation (handle); (* Bildschirm loeschen *)
175:   Points [0] := 0; Points [1] := 0;
176:
177:   MaxZaehler := cMaxZaehler;
178:   REPEAT
179:     LoescheBild; CursorEin;
180:     WriteString ('G(enerieren, D(atei, Z(aehler einstellen (');
181:     WriteCard (MaxZaehler, 1);
182:     WriteString ('), E(nde: ');
183:     Read (Befehl); WriteLn; WriteLn;
184:     CASE CAP (Befehl) OF
185:
186:       (* Neues Bild erzeugen: *)
187:
188:       'G':      ClearWorkstation (handle);
189:                 WriteString ('Bilddatei (nur RET keine): ');
190:                 ReadString (Dateiname); WriteLn;
191:                 IF Length (Dateiname) # 0 THEN
192:                   OpenStream (Datei, Dateiname, READWRITE, ok);
193:                   Protokoll := TRUE;
194:                 ELSE Protokoll := FALSE;
195:                 END (* IF Dateiname *);
196:                 WriteString ('Linke untere Ecke (x): ');
197:                 ReadReal (Ecke.r); WriteLn;
198:                 WriteString ('(y): ');
199:                 ReadReal (Ecke.i); WriteLn;
200:                 WriteString ('Seitenlaenge: ');
```

```
201:                ReadReal (Seite); WriteLn;
202:                CursorAus;
203:                ClearWorkstation (handle);
204:
205:                Spalt := Seite / FLOAT (cSpalten);
206:
207:                alteFarbe := weiss;
208:
209: (*************************************************************************)
210: (* Dies ist der eigentliche Algorithmus, der die Mandelbrot-Menge
211:    bestimmt: *)
212:                FOR m := 0 TO cZeilen DO
213:                  FOR n := 0 TO cSpalten DO
214:                    c.r := Ecke.r + FLOAT (n) * Spalt;
215:                    c.i := Ecke.i + FLOAT (m) * Spalt;
216:                    z.r := 0.0; z.i := 0.0;
217:                    Zaehler := 0;
218:                    IF (c.r > -0.5) & (c.r < 0.25) &
219:                       (ABS (c.i) < 0.375) THEN
220:                       (* Im Inneren rechteckiges Feld ueberspringen *)
221:                       Zaehler := MaxZaehler + 1
222:                    ELSE
223:                      REPEAT (* Iteration *)
224:                        WITH z DO
225:                          k.r := r * r - i * i + c.r;
226:                          k.i := 2.0 * r * i + c.i;
227:                          z := k;
228:                        END (* WITH *);
229:                        INC (Zaehler);
230:
231:                        (* Write ('.'); *)
232:
233:                      UNTIL (Betrag (z) > 2.0) OR (Zaehler > MaxZaehler);
234:                    END (* IF *);
235:
236:                    Farbe (Zaehler, n, m); (* Pixel setzen *)
237:
238:                     (* Ev. Matrixwert in Datei schreiben: *)
239:
240:                    IF Protokoll THEN
241:                      Write8Bit (Datei, CHR (Zaehler));
242:                    END (* IF Protokoll *);
243:
244:                  END (* FOR n *);
245:                END (* FOR m *);
246:
247: (*************************************************************************)
248:
249:                IF Protokoll THEN CloseStream (Datei, ok); END;
250:                Read (ch);
251:
252:
253:       (* Altes Bild von Diskette darstellen: *)
254:
255:        D':     WriteString ('Name der Bilddatei: ');
256:                ReadString (Dateiname); WriteLn;
257:                IF Length (Dateiname) # 0 THEN
258:                  OpenStream (Datei, Dateiname, READ, ok);
259:                  IF ok = 0 THEN
260:                    CursorAus;
261:                    ClearWorkstation (handle);
262:                    FOR m := 0 TO cZeilen DO
263:                      FOR n := 0 TO cSpalten DO
264:                        Read8Bit (Datei, ch);
265:                        Zaehler := ORD (ch);
266:                        Farbe (Zaehler, n, m);
267:                      END (* FOR n *);
268:                    END (* FOR m *);
269:                    CloseStream (Datei, ok);
270:                    Read (ch);
271:                  ELSE WriteString ('Datei nicht gefunden!');
272:                  END (* IF Datei nicht gefunden *);
273:                END (* IF Dateiname *);
274:
275:
276:       (* Iterationszaehler einstellen: *)
277:
278:        Z':     WriteString ('Neuer Zaehlerwert: ');
279:                ReadCard (MZ);
280:                MaxZaehler := MZ;
281:                ;
282:       ELSE;
283:     END (* CASE *);
284:   UNTIL CAP (Befehl) = 'E';
285:   ok := SetLineColour (handle, schwarz);
286:   CursorAus;
287: END Apfelmaennchen.
288:
```

Listing des vollständigen Apfelmännchen-Programms (Schluß)

Apfelmännchen

Jeder kennt inzwischen »Apfelmännchen«. Aber wer weiß, was dahintersteckt?

In den letzten Monaten lebte die halbe Computerwelt im Fractal-Fieber. Leider Gottes blieb dabei eines auf der Strecke, nämlich welche Idee hinter diesen Bildern steht und was sie aussagen. Wir haben nicht die Absicht, den 1001-ten Apfelmännchen-Algorithmus zur Abwechslung als Flußdiagramm zu liefern, sondern allen, die an der dahinterstehenden Mathematik interessiert sind, etwas Klarheit zu verschaffen.

Der durch die Fractal-Grafiken dargestellte mathematische Sachverhalt geht ziemlich ans Eingemachte. Wir erklären deshalb zunächst einige Begriffe. Diese Definitionen sind vereinfacht und nur in dem nötigen Umfang durchgeführt und erheben keinen Anspruch auf mathematische Vollständigkeit. Wer hier detaillierteres Wissen erlangen will, sei auf die angegebene Literatur verwiesen.

Als erstes widmen wir uns den für die Fractals bedeutsamen komplexen Zahlen. Dazu ist es sinnvoll, zuerst die reellen Zahlen zu betrachten. Zu deren Veranschaulichung dient der sogenannte Zahlenstrahl (Bild 1). Jedem Punkt auf der Geraden ist genau eine reelle Zahl zugeordnet. Stellt man sich die Gerade in beide Richtungen unendlich weit fortgesetzt vor, so erhält man die Menge der reellen Zahlen.

Eine komplexe Zahl z besteht nun im Gegensatz zu einer reellen Zahl nicht nur aus einer Komponente, zum Beispiel 271,35754, sondern aus einem geordneten Paar zweier reeller Zahlen wie 2,59 und −2,1, mit der Schreibweise z(2,59,-2,1). So ergibt sich eine Ebene (Bild 2). Dabei bezeichnet man die eine Achse als die reelle (Re), die andere als die imaginäre Achse (Im) und bezeichnet die entsprechenden Komponenten der Zahl z als Real- (2,59) und als Imaginärteil (−2,1).

Wahrscheinlich ist Ihnen inzwischen an Hand der beiden Bilder aufgefallen, daß die reellen Zahlen eine Art Sonderfall sind. Nämlich die Menge aller komplexen Zahlen mit dem Imaginärteil 0.

Will man jetzt mit den komplexen Zahlen rechnen, wählt man die sogenannte Normalform: $z(x;y) = x + iy$, wobei x der Realteil und y der Imaginärteil von z ist, und das i lediglich zur Kennzeichnung des Imaginärteils dient. Dabei gilt die folgende Festlegung:

$$i \cdot i = -1 \qquad (1)$$

Bei der Addition zweier komplexer Zahlen u(x1;y1) und v(x2;y2) erhält man durch die Normalform:

$$u(x1;y1) + v(x2;y2) =$$
$$(x1 + iy1) + (x2 + iy2) =$$
$$x1 + x2 + iy1 + iy2 =$$
$$(x1 + x2) + i(y1 + y2) =$$
$$z((x1 + x2); i(y1 + y2)) \qquad (2)$$

Für die Multiplikation ergibt sich das Folgende:

$$u(x1;y1)v(x2;y2) =$$

(x1 + iy1) (x2 + iy2) =
x1x2 + x1iy2 + iy1x2 + iy1iy2 =
x1x2 + iiy1y2 + ix1y2 + ix2y1 =
nach (1)
(x1x2 – y1y2) + i(x1y2 + x2y1) =
z((x1x2 – y1y2);(x1y2 + x2y1)) (3)

Zuletzt noch zum Betrag IZI einer komplexen Zahl z(x;y):

$$|Z| = \sqrt{x^2 + y^2} \quad (4)$$

der die Länge der Strecke o(0;0) z(x;y) angibt, also die »Entfernung« der Zahl z vom komplexen Ursprung.
Betrachtet man alle komplexen Zahlen z mit gleichem Betrag IZI, so stellt man fest, daß diese auf einem Kreis mit Radius IZI um den komplexen Ursprung liegen (Bild 3).

Damit sind wir auch schon bei den Folgen. Unter einer endlichen Folge versteht man eine endliche Menge von reellen beziehungsweise komplexen Zahlen, denen man eine feste Reihenfolge zugeordnet hat. Geht man von einer 6-elementigen Menge reeller Zahlen [0,31;0;–2; 5; 521; 2,7] aus und ordnet man ihnen die feste Reihenfolge $a_1 = 5$; $a_2 = 2{,}7$; $a_3 = 521$; $a_4 = 0{,}31$; $a_5 = 0$; $a_6 = -2$ zu, so erhält man eine reelle Folge der a_n. Die a_n bezeichnet man dabei als Glieder der Folge a_n.

Wenn man zur Berechnung des nächsten Folgengliedes eines der vorhergegangenen Glieder heranzieht, dann nennt man eine solche Folge rekursiv (= auf sich selbst gestützt). Beispiel: $a_1 = 1$; $a_{n+1} = a_n + {}^1/_{n^2}$ für n > 1 hat die Folge die Glieder: 1; $^3/_9$; $^{11}/_6$; $^{25}/_{12}$;...

So kann man auch eine endliche Menge von Werten einer Funktion als endliche Folge auffassen, zum Beispiel $a_n = 1/n^2$ für n = 1; 2; ... 10 liefert die Folge: 1; $^1/_4$; $^1/_9$; $^1/_{16}$; $^1/_{25}$; $^1/_{36}$; $^1/_{47}$; $^1/_{64}$; $^1/_{81}$; $^1/_{100}$. Betrachtet man jetzt die einzelnen Folgenglieder weiter, auch für größeres n, so stellt man fest, daß die Werte kleiner werden und sich der Null annähern.

Und damit sind wir schon bei den beiden letzten Definitionen: Konvergenz und Grenzwert. Hat eine Folge die Eigenschaft, daß man einen Index N finden kann, ab dem alle weiteren Folgenglieder a_n, n > N, beliebig nahe an einem Wert a liegen, so konvergiert die Folge gegen den Wert a, und man bezeichnet diesen als den Grenzwert a der Folge a_n.

In unserem Beispiel (Bild 1): gleichgültig, wie klein man das Intervall I um 0 auch macht, man kann stets ein N finden, so daß für alle a_n mit n > N der Wert von $1/n^2$ innerhalb des Intervalls I liegt. Wählt man sich das Intervall I = [- 0,01; 0,01], so muß N = 10 gewählt werden, damit für alle n > N der Wert von $a = 1/n^2$ im ausgewählten Intervall I liegt.

Sofern eine Folge diese Eigenschaft nicht hat (zum Beispiel $a_n = n^3$), dann bezeichnet man sie als divergent, und es existiert kein Grenzwert a. Was hier über die reellen Folgen dargelegt worden ist, gilt im wesentlichen auch für die komplexen Folgen. Der Unterschied besteht nur darin, daß für die Konvergenz ein Index N zu finden sein muß, damit für alle n > N die Folgenglieder a_n innerhalb eines Kreises mit beliebig kleinem Radius um den Grenzwert liegen (Bild 4).

Augen schließen – einmal kräftig durchatmen – geschafft, aus ist es mit der grauen Theorie! »Und was hat das nun mit unserem schönen Apfelmännchen zu tun?« werden sich wohl viele jetzt entsetzt fragen. Die Antwort ist, sobald man sich durch den mathematischen Dschungel gekämpft hat, verhältnismäßig einfach: Die Fractal-Grafiken veranschaulichen das Konvergenzverhalten von rekursiven komplexen Folgen.

Zur Erzeugung dieser Bilder wählt man einen zu untersuchenden rechteckigen Ausschnitt aus der komplexen Zahlenebene aus. Dann wird die Breite des Ausschnitts durch die X-Auflösung und die Höhe durch die Y-Auflösung des Rechners dividiert, wodurch jedem Bildschirmpunkt eine komplexe Zahl c aus dem Ausschnitt zugeordnet wird (Bild 5). Dann folgt für jeden Bildschirmpunkt einer rekursiven komplexen Folge die Berechnung. Beim Apfelmännchen lautet diese $z_{n+1} = z_n^2 + c$ mit $z_1 = c$, wobei c die dem Bildschirmpunkt zugeordnete komplexe Zahl ist. Bei der Berechnung der Folge wird mitgezählt (Laufindex n), wie viele Versuche notwendig sind, damit der Betrag des letzten Folgengliedes z_n einen fest vorgegebenen Schwellwert überschreitet (beim Apfelmännchen zum Beispiel 100). Diese Anzahl von rekursiven Gliederberechnungen wird dann als Farbpunkt auf dem Bildschirm ausgegeben. Gelingt die Schwellwertüberschreitung in einer festgelegten Anzahl von Versuchen nicht, wird ein schwarzer Punkt gesetzt.

Anhand einer so erzeugten Grafik kann man nun schnell das Konvergenzverhalten der untersuchten Folge feststellen: Für die Werte von c, für die sich ein farbiger Punkt auf dem Bildschirm ergibt, divergiert die Folge, und es läßt sich anhand des Farbwertes feststellen, wie schnell sie das tut.

Die Bilder stellen nur eine Idealisierung der Konvergenz dar, da diese ja eigentlich nur für eine unendliche Anzahl von Folgegliedern sinnvoll ist. Da der Rechner aber nicht in der Lage ist, unendlich viele Folgeglieder pro Punkt zu ermitteln, idealisiert man die Betrachtung auf nur endlich viele Folgeglieder. Damit ist es einleuchtend, daß man mit einer höheren maximalen Rekursionstiefe näher an die Wirklichkeit herankommt.

Um den Lesern einen rechnerunabhängigen Eindruck von der Arbeitsweise eines Fractal-Programms geben zu können, zeigen wir hier ein Flußdiagramm.

Wir verzichteten auf Routinen wie zum Beispiel solche für die Eingabe der Parameter »Linke Grenze«, »Rechte Grenze«, »Schwelle« und »Tiefe«, da sich diese sicher jeder selbst schreiben kann. Das Unterprogramm »Zeichnen« berechnet zuerst die Schrittweite von einem Parameter c zum nächsten. Deren Realteil ergibt sich durch Division der Breite des Ausschnitts durch die x-Auflösung, der Imaginärteil durch Division der Höhe des Ausschnitts durch die Y-Auflösung des jeweiligen Computers. So wird jedem Bildschirmpunkt ein Parameter c zugewiesen. In den folgenden Schleifen errechnet ein Unterprogramm (Punkt) den für jeden Bildschirmpunkt zugehörigen Folgeparameter c. Dazu multipliziert es die vorher im Hauptprogramm ermittelte Schrittweite mit dem X-Wert und addiert den Wert der linken Grenze des komplexen Ausschnitts dazu. So erhält man die zum Bildschirmpunkt X,Y zugehörige komplexe Zahl c. Ein weiteres Unterprogramm berechnet die rekursive Folge. Als Ergebnis wird die Anzahl der notwendigen rekursiven Berechnungen zurückgegeben, also der Farbwert des zu setzenden Punktes. Daraufhin wird der Punkt in der so ermittelten Farbe auf den Koordinaten X, Y gesetzt.

Unterprogramm »Folge«: Zunächst wird die Farbe zurückgesetzt. Danach wird der Startwert z, also c zugewiesen.

Dann erfolgt die eigentliche Berechnung der rekursiven Folge. In einer Hilfsvariablen werden ausgehend von dem jeweils vorhergegangenen Wert der Real- und Imaginärteil des neuen Folgegliedes berechnet, die dann an die dafür vorgesehenen Variablen zugewiesen werden.

Nach Ermittlung des Betrags des aktuellen Folgegliedes und Erhöhung des Farbwerts erfolgt die Überprüfung auf das Ende der Berechnung. Sofern entweder die Rekursionstiefe oder der Schwellwert überschritten wurde, ist die Berechnung beendet. Zuletzt wird, wenn die Rekursionstiefe überschritten wurde, der kleinste Farbwert der Variable Farbe zugewiesen.

Eine weitere Möglichkeit besteht darin, einen anderen Punkt als den komplexen Ursprung als Konvergenz-Zentrum zu betrachten.

Auch lassen sich ganz andere Konvergenzvorgänge wie zum Beispiel die Konvergenz von Nullstellenverfahren für Polynome oder ähnliche Sachverhalte durch solche Grafiken untersuchen. (Marten Moser/hb)

Bild 1. Zahlenstrahl

Bild 2. Eine komplexe Zahlenebene

Bild 3. Punkte in einem Kreis mit komplexem Ursprung

Bild 4. Konvergenzen einer komplexen Folge a;(x)

Bild 5. Zuordnung von komplexen Zahlen zu Bildschirmpunkten

UP-Zeichen
Schrittweite re = (Rechte Grenze re - linke Grenze) : X-Auflösung
Schrittweite i = (Rechte Grenze im - linke Grenze) : Y-Auflösung
X = 0
Y = 0
UP-Punkt
Y = Y + 1
Y < Y-Auflösung?
X = X + 1
X < X-Auflösung?
ENDE

Farbe > Tiefe?
Nein
Farbe = kleinster Farbwert
Ende UP

UP-Punkt berechnen
C re = Schrittweite re * X + Linke Grenze re
C im = Schrittweite im * Y + Linke Grenze im
UP-Folge
Farbe = Farbe Modulo (Größter Farbwert - 1) + 1
Punkt auf X, Y in Farbe setzen
Ende UP

UP-Folge berechnen
Farbe = kleinster Farbwert
Z re = C re
Z im = C im
h re = Z re * Z re - Z im * Z im + C re
h im = 2 * Z re * Z im + C im
Z re = h re
Z im = h im
Betrag = Z re * Z re + Z im * Z im
Farbe = Farbe + 1
Farbe > Tiefe oder Betrag > Schwelle?

Bild 6. Flußdiagramm zum Apfelmännchen

Grafik-

Herrlich einfach ist es, in Assembler eigene Grafik-Routinen für den Atari ST zu entwerfen. Wir zeigen, wie's geht.

Der Atari ST besitzt ein umfangreiches Grafik-Paket im TOS (Tramiel Operating System). Trotzdem ist es überaus interessant, sich seine eigenen Grafik-Routinen zu programmieren. Man kann sie individuellen Vorstellungen anpassen. Zum Beispiel eine Line-Routine definieren, die keine gerade Linie zieht, sondern an den gewünschten Stellen einen Kreis malt.

Die abgedruckten Routinen sollen als Anstoß zu derartigen Experimenten dienen. Den Anfang bildet eine kleine Demo-Routine, die einige schöne Effekte auf dem Bildschirm zaubert.

Doch nun einiges zu diesen Grafikroutinen, die vollständig in 68000-Assembler geschrieben sind. Entstanden sind sie auf dem GST-Assembler Dieser Assembler weist keine großen Unterschiede zu anderen 68000-Assemblern auf, so daß eine Adaption auf andere Versionen kein Problem ist.

Die erste Routine, »setscreen«, dient dem Setzen verschiedener Bildschirmparameter: Die physikalische Startadresse des Grafik-RAM steht im 68000-Register D0. Sie gibt an, was gezeigt werden soll. Die logische Adresse des Grafik-RAM im Register D1 legt fest, an welcher Stelle gezeichnet werden soll. Den Grafikmodus enthält D2 (0 = 640 x 400, 1 = 640 x 200, 2 = 320 x 200). Die beiden Adressen werden im Register wie üblich im Langwort-Format (4 Bytes) gesetzt. Im Grafikmoduswert geschieht dies als Byte. Möchte man beim Aufruf der Routine bestimmte Register unverändert lassen, so lädt man das Register mit einem negativen Wert (Bit 31 beziehungsweise 7 gesetzt).

Interessant bei der Programmierung dieser Routine ist die Verwendung der GEMDOS-Funktion »SUPER«. Dadurch aktiviert man den Supervisor-Modus des 68000. Das ist notwendig, da im User-Modus kein Zugriff auf den I/O-Bereich des ST möglich ist. Ein Versuch würde zu einem Busfehler führen.

Die Routine »hole__screenparameter« dient zum Kopieren der physischen und logischen Startadresse des Grafikspeichers. Die folgenden Grafikroutinen bedienen sich häufig dieser Kopien.

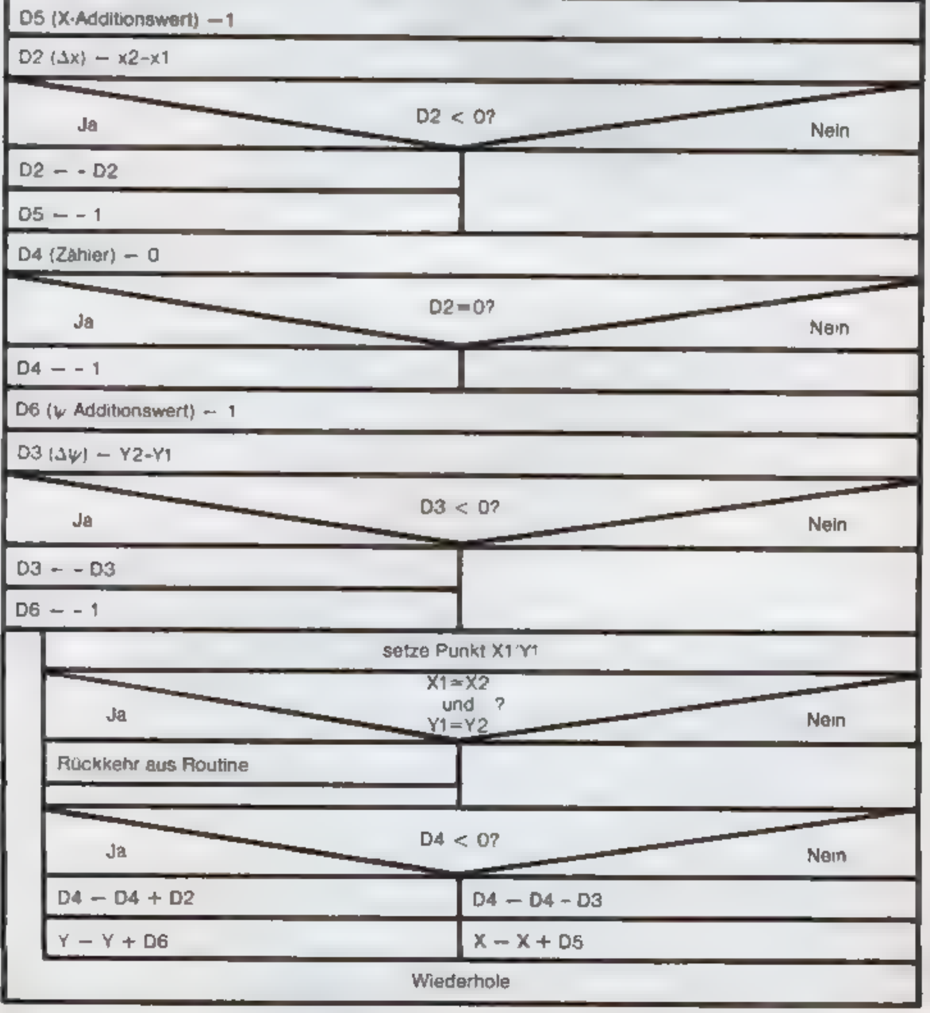
»setze__farben« belegt die Farbregister mit den gewünschten Werten, die in den nachfolgenden Hexdaten (im Wort-Format) stehen.

»screen__init« ruft die Routinen »setscreen«, »setze__farben«, »hole__screenparameter« und »screen__clear« auf und schaltet dabei den 320 x 200-Modus ein. Die Bildschirmadressen selbst bleiben dabei unverändert. Zudem wird der Cursor, der sonst ungestüm Grafik kaputtblinken würde, abgeschaltet.

An dieser Stelle zwei wichtige Hinweise: Programme, die die nachfolgenden Grafikroutinen verwenden, müssen als TOS-Anwendung angemeldet, also mit der Endung ».TOS« versehen sein. Ansonsten kommt es zu Konflikten mit dem Maus-Cursor. Außerdem sollten sie immer vom 320 x 200er-Modus gestartet werden, da es sonst passieren kann, daß man nach Ende des Programms vor einem arg verkrüppelten Desktop steht (die entsprechenden GEM-Parameter für die Auflösungsstufe werden von diesem Programm nicht gestellt).

»screen__clear« löscht die Bildschirmseite ab dem logischen Grafikstart.

Die folgende »plot«-Routine wurde speziell für den 320 x 200-Grafikmodus geschrieben. In den Registern D0 (für x) und D1 (für y) übergibt man die Koordinaten des zu setzenden/testenden Punktes (Länge: Wort), Ursprung (0/0) ist die linke obere Bildschirmecke. Liegen die Koordinaten außerhalb, beendet das die Routine vorzeitig mit einer Fehlermeldung. Die gewünschte Farbe wird in die Speicherstelle



Struktogramm zur Line-Routine

Routinen

»color__plot« geschrieben. Dabei werden nur die Bits 0 bis 3 berücksichtigt (nur Farbwerte von 0 bis 15). Enthält diese Speicherstelle einen negativen Wert (Bit 7 = 1), so holt sie die Farbe des entsprechenden Punktes und speichert ihn in »color__plot«.

Ein Aufruf der »plot«-Routine ändert die Werte der Register D0 bis D3 und A0. Falls diese wichtig sind, sollten sie vor dem Funktionsaufruf mit dem 68000-Befehl »MOVEM« gerettet werden. Mehr Informationen über die Programmierung der Plot-Routine entnehmen Sie bitte dem Struktogramm.

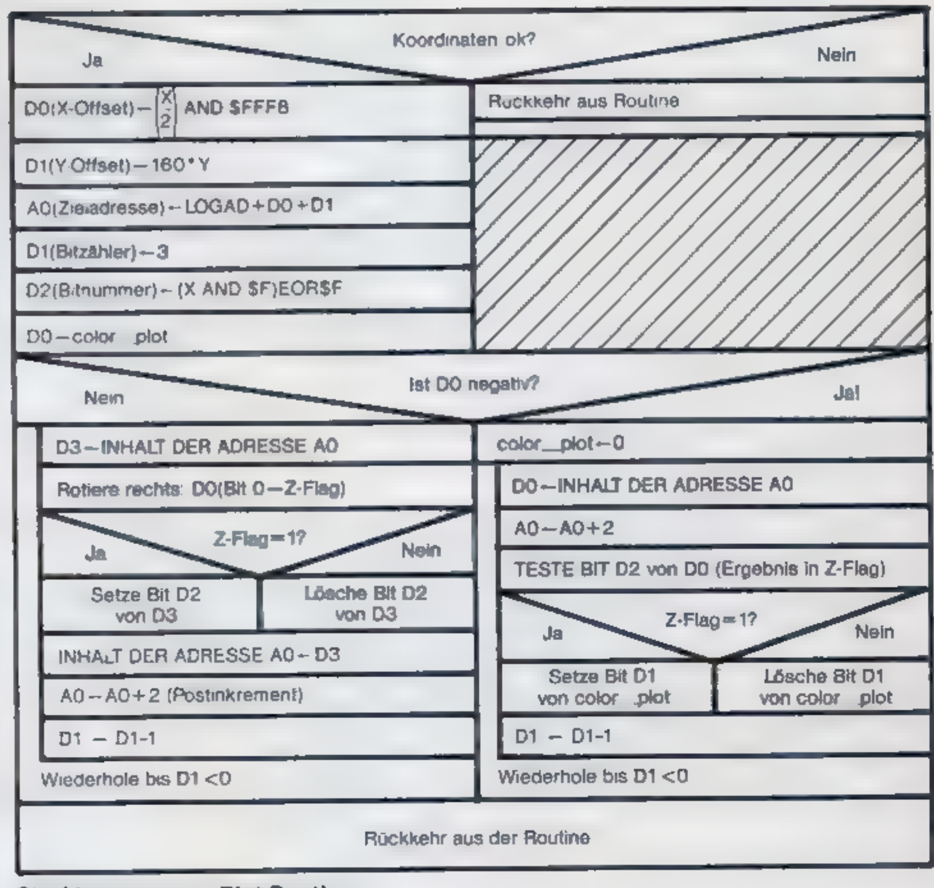
Von Punkt zu Punkt

»line« ist eine Routine zum Ziehen einer Linie. Diese Routine kann man auch in den anderen Grafikauflösungen verwenden, wenn man mit einer anderen Plot-Routine arbeitet. Die Farbe wird in »color__plot« gespeichert (Byte), die Koordinaten stehen im Format »Wort« in folgenden Registern: X1 ist D0, Y1 ist D1, X2 ist D2, Y2 ist D3. Sämtliche von der Line-Routine verwendeten Register werden gerettet und bleiben somit erhalten. Mehr Details über die Programmierung der Line-Routine vermittelt das Struktogramm.

»hoch__scroll« und »runter__scroll« rollen den HiRes-Schirm ab der logischen Startadresse jeweils um eine Zeile nach oben oder unten. Sämtliche Inhalte der verwendeten Register werden gerettet und sind nach dem Aufruf weiterhin vorhanden. Wie gesagt, sollen diese Routinen nur als Startpaket dienen und individuellen Anwendungen angepaßt werden. Viel Spaß beim Experimentieren! (Frank Mathy/hb)

Programmierer an den ST

Nehmen Sie dieses Programmbeispiel zum Anlaß, weitere Routinen zu entwickeln. Senden Sie uns Ihr Programm auf Diskette und als Hardcopy zu. Bitte vergessen Sie nicht eine ausführliche Programmerklärung beizulegen. Bei Rückfragen wenden Sie sich bitte an unseren Redakteur: H. Brandl. Telefon 089/4613-336.



Struktogramm zur Plot-Routine

```
                                                          A:\HIRES1.ASM

 SECTION S.GLOB

*======================================================================
* Demoroutine zum Hirespac
*======================================================================

start           bsr     screen_init      *Bildschirm initialisieren
                move.w  #200,d0          *Erste Line-Kooordinaten setzen
                move.w  #47,d1
                move.w  #120,d2
                move w  #167,d3
                move.w  #1,color_plot    *Erste Farbe setzen
                move.w  #1,x1_richtung   *Bewegungsregister setzen
                move.w  #1,x2_richtung
                move.w  #1,y1_richtung
                move.w  #1,y2_richtung
st_loop         bsr     line             *Linie zeichnen
                add.b   #1,color_plot    *Farbe erhöhen
                and.b   #$f,color_plot   *Bits 4-7 ausblenden
                add.w   x1_richtung,d0   *Neue Punkte ermitteln
                add w   y1_richtung,d1
                add.w   x2_richtung,d2
                add.w   y2_richtung,d3
                cmp.w   #319,d0          *Stoßen Punkte an?
                bcs.s   st_ok1
                bsr     runter_scroll    *Bildschirm runterscrollen
                neg.w   x1_richtung      *Richtung negieren
st_ok1          cmp.w   #319,d2
                bcs.s   st_ok2
```

Listing der Grafik-Routinen für den Atari ST

```
                bsr       runter_scroll
                neg w     x2_richtung
st_ok2          cmp w     #199,d1
                bcs s     st_ok3
                bsr       runter_scroll
                neg w     y1_richtung
st_ok3          cmp.w     #199,d3
                bcs.s     st_ok4
                bsr       runter_scroll
                neg w     y2_richtung
st_ok4          move.w    d0,-(sp)          *D0 sichern
                move w    #$ff,-(sp)        *liegt Tastendruck vor?
                move.w    #6,-(sp)          *GEMDOS-Funktion 6 >>rawconio<<
                trap      #1
                addq.l    #4,sp             *Stack reparieren
                move w    d0,d4             *Ergebnis kopieren
                move w    (sp)+,d0          *D0 zurückholen
                tst.w     d4                *Ergebnis testen
                beq       st_loop           *Kein Tastendruck -> Schleife
                move w    #1,-(sp)          *GEMDOS-Funktion 1
                trap      #1                *Aufrufen Auf Taste warten
                addq.l    #2,sp             *Stack in Ordnung bringen
                clr w     -(sp)             *GEMDOS-Fkt. 0 TERM
                trap      #1                *Beenden des Programmes

x1_richtung     dc.w      1
x2_richtung     dc.w      1
y1_richtung     dc.w      1
y2_richtung     dc.w      1

**********************************************************************
* Hires-Routinen für den Atari ST
* Die Plot/Pixelabfrage-Routine ist speziell für den
* 320x200 Grafikmodus
* geschrieben, die anderen (auch Line, wenn eine andere Plot-Routine
* aufgerufen wird) laufen auch in den anderen Modi
*
* Geschrieben von       Frank Mathy         (c) Januar 1986
*                       H.-Heine-Str. 20, 6200 Wiesbaden 13
**********************************************************************

* Setzen der Farbpalette

setze_farben    move.l    #farbpalette,-(sp)        *Adresse der Farbpalette
                move.w    #6,-(sp)                  *XBIOS-Befehl setpalette
                trap      #14                       *Ausführen des Befehls
                addq.l    #6,sp                     *Stackpoint. korrigieren
                rts                                 *Rückkehr

farbpalette     dc w      $ffff,$ff88,$f8f8,$fda8,$f88f,$f8d8,$f68d,$fddd
                dc w      $faaa,$f8ff,$f8dd,$fff8,$ff8f,$fdd8,$fd8d,$f888

* Setzen der Bildschirmparameter:
* Auflösung, physikalische und logische Bildschirmadresse

* Grafikmodi :  0 = 320*200 Pixels bei 16 Farben
*               1 = 640*200 Pixels bei 4  Farben
*               2 = 640*400 Pixels bei 2  Farben
*              -1 = nicht verändern

* Adressen:     Wenn zu setzen, dann 4-Byte-Adresse
*               Wenn nicht zu setzen, dann -1

* Bei Aufruf :  D0 enthält physikalische Adresse/-1 (Longword-4 Bytes)
*               D1 enthält logische      Adresse/-1 (    "         "  )
*               D2 enthält Grafikmodusbefehl           (Byte)
screenpt        equ       $45e
v_bas_ad        equ       $44e
sshiftmd        equ       $200

setscreen       tst.l     d0                *Wird physik. Adr. geändert?
                bmi s     setb_1            *Nein -> weiter
                move.l    d0,physad         *Kopie der physik. Bildschirmadr.
                move l    d0,screenpt       *Bei nächstem VBL setzen
setb_1          tst l     d1                *Wird log. Adr. geändert?
                bmi s     setb_2            *Nein -> weiter
                move l    d1,logad          *Kopie der log. Bildschirmadr.
                move.l    d1,v_bas_ad       *Entspr. Systemvariable stellen
setb_2          tst b     d2                *Wird Auflösung geändert?
                bmi s     setb_3            *Nein -> weiter
                clr.l     -(sp)             *Superv.-Stack-Pt =User-Stackpt.
                move.w    #$20,-(sp)        *BIOS-Funktion $20 S[illegible]
                trap      #1                *BIOS-Aufruf
                add.l     #6,sp             *Stackpointer reparieren
```

Listing der Grafik-Routinen für den Atari ST

```
                 move b  d2,$ff8260      *Auflösung setzen
                 move b  d2,sshiftmd     *Entspr. Systemvariable
                 move l  d0,-(sp)        *alter Supervisor-Stackpointer
                 move w  #$20,-(sp)      *Befehl Super
                 trap    #1              *Aufruf
                 add.l   #6,sp           *Stackpointer in Ordnung bringen
setb_3           rts                     *Das wär's!

physad           ds.l    1               *Kopie der phys. Bildschirmadr.
logad            ds.l    1               *Kopie der log.  Bildschirmadr.


* Holen der physischen und logischen Bildschirmadresse und kopieren der
* Adressen in physad und logad, da auf diese im User-Mode zugegriffen
* werden kann

hole_scrnparam   move.w  #2,-(sp)        *Befehl physbase
                 trap    #14             *Aufrufen des XBIOS
                 move.l  d0,physad       *Adresse kopieren
                 move.w  #3,-(sp)        *Befehl logbase
                 trap    #14             *Aufruf
                 move.l  d0,logad        *Adresse kopieren
                 addq.l  #4,sp           *Stackpointer reparieren
                 rts                     *Ende der Routine

* Initialisieren aller verwendeten Parameter und Einstellungen
* Sollte beim Programmstart aufgerufen werden

screen_init      move b  #0,d2
                 move.l  #-1,d0
                 move l  #-1,d1
                 bsr     setscreen
                 bsr     setze_farben
                 bsr.s   hole_scrnparam  *Kopien der Adressen
                 move.w  #0,-(sp)        *Cursur ausschalten
                 move.w  #21,-(sp)       *XBIOS-Funktion cursconf
                 trap    #14
                 addq.l  #4,sp           *Stackpointer zuücksetzen
                 bsr.s   screen_clear    *Löschen des Bildschirms
                 rts

* Löschen des Bildschirms (logische Bildschirmadresse)

screen_clear     move.w  #7999,d0        *8000 Langwörter zu löschen
                 movea.l logad,a0        *Pointer
scr_clr_loop     clr.l   (a0)+           *Löschen und Pointer erhöhen
                 dbne    d0,scr_clr_loop *Schleife
                 rts                     *Alles sauber..

* Plot-Routine, X/Y-Koordinaten in D0/D1(Wort),
* Farbe in color_plot (Byte)
* Wenn color plot negativ (>$7f) dann wird die Farbe des entsprechenden
* Punktes auf der logischen Bildschirmseite geholt
* Die Farbe steht dann in color_plot (Byte)

plot             cmp.w   #320,d0         *x-Koordinate in zugel. Grenze?
                 bcc.s   plot_ende       *Nein -> Ende
                 cmp.w   #200,d1         *y-Koordinate in zugel. Grenze?
                 bcc.s   plot_ende       *Nein -> Ende
                 move w  d0,d2           *x-Koordinate retten
                 lsr.w   #1,d0           *d0=d0/2
                 and w   #$fff8,d0       *Bits 0-2 löschen
                 and.l   #$0000ffff,d0   *Damit Bits 16-31 auf 0
                 mulu.w  #160,d1         *Y-Offset bestimmen:yoff=160*d1
                 add l   d1,d0           *d0=d0+y-Offset
                 add l   logad,d0        *Nun haben wir Zieladresse
                 movea.l d0,a0           *Ins Adressregister damit
                 move w  #3,d1           *Bitzähler
                 and w   #$0f,d2         *Bits 0-3 geben an, welches Bit
                 eor w   #$0f,d2         *Invertieren wegen Reihenfolge
                 move.b  color_plot,d0   *Farbe in d0
                 bmi s   plot_get
plot_loop        move.w  (a0),d3
                 lsr.b   #1,d0           *akt. Bit 0 d. Farbe ins Carry
                 bcs s   plot_1          *Branchen, wenn Carry=1
                 bclr.l  d2,d3           *Entspr. Bit löschen
                 bra s   plot_2          *Weiter
plot_1           bset.l  d2,d3           *Entspr. Bit setzen
plot_2           move.w  d3,(a0)+        *Zurückschreiben
                 dbra    d1,plot_loop    *Erniedrigen und ggf. Schleife
plot_end         rts

* Farbe eines Pixels holen

plot_get         clr.b   color_plot      *Farbregister reinigen
```

Listing der Grafik-Routinen für den Atari ST (Fortsetzung)

```
pl_get_1        move.w  (a0)+,d0
                btst.l  d2,d0            *Bit testen
                bne s   pl_get_2         *Gesetzt -> Verzweigen
                bclr.b  d1,color_plot    *Bit löschen
                bra.s   pl_get_1         *Weiter an anderer Stelle
pl_get_2        bset.b  d1,color_plot    *Bit setzen
pl_get_3        dbra    d1,pl_get_1      *Erniedrigen und ggf. Schleife
                rts                      *Fertig


color_plot      ds.b    1

* Line-Routine, X1/Y1/X2/Y2 in D0/D1/D2/D3 (Wort)

line            movem.l d0-d7/a0,-(sp)   *Register in Sicherheit bringen
                move.w  d2,line_xend     *X2 retten
                move.w  d3,line_yend     *Y2 retten
                move.w  #1,d5            *X-Additionswert=1
                sub.w   d0,d2            *Delta X = X2-X1
                bpl.s   line_dxpos       *Delta X positiv -> weiter
                neg.w   d2               *Delta X negieren (dann positiv)
                move.w  #-1,d5           *X-Additionswert=-1
line_dxpos      clr.w   d4               *d4 ist der Zähler
                tst.w   d2               *Delta X nochmal testen
                bne.s   lin_dxnnull      *Delta X <> 0 -> weiter
                move.w  #-1,d4           *Zähler auf -1
lin_dxnnull     move.w  #1,d6            *Y-Additionswert=1
                sub.w   d1,d3            *Delta Y = Y2-Y1
                bpl.s   line_loop        *Delta Y positiv -> weiter
                neg.w   d3               *Delta Y negieren (dann positiv)
                move.w  #-1,d6           *Y-Additionswert=-1
line_loop       movem.w d0-d3,-(sp)      *Register retten
                bsr     plot             *Punkt setzen
                movem.w (sp)+,d0-d3      *Register zurückholen
                cmp.w   line_xend,d0     *Letzter x-Wert?
                bne.s   line_weiter      *Nein -> weiter
                cmp.w   line_yend,d1     *Auch letzter y-Wert?
                bne.s   line_weiter      *Nein -> weiter
                movem.l (sp)+,d0-d7/a0   *Alte Registerwerte wiederherst.
                rts                      *Line fertig, Rückkehr
line_weiter     tst.w   d4               *Testen des Zählers
                bmi.s   linneg           *Negativ -> Branch
                sub.w   d3,d4            *Zähler=Zähler-Delta y
                add w   d5,d0            *X = X + x Additionswert
                bra.s   line_loop
linneg          add.w   d2,d4            *Zähler=Zähler+Delta x
                add.w   d6,d1            *Y = Y + y Additionswert
                bra.s   line_loop

line_xend       ds.w    1
line_yend       ds.w    1

* Soft-Scroll: Diese Routinen rollen den Bildschirm je um eine Bild-
* schirmzeile nach oben bzw. unten

hoch_scroll     movem.l d0/a0-a1,-(sp)
                movea.l logad,a0         *hierein wird kopiert
                movea.l logad,a1         *hieraus wird kopiert
                add.l   #160,a1          *deshalb muß a1 größer sein
                move.w  #7959,d0         *so oft muß kopiert werden
sc_h_1          move.l  (a1)+,(a0)+      *kopieren
                dbra    d0,sc_h_1        *Zähler erniedrigen, ggf. Schleife
                move.w  #39,d0           *so oft muß gelöscht werden
sc_h_2          clr.l   (a0)+            *löschen
                dbra    d0,sc_h_2
                movem.l (sp)+,d0/a0-a1
                rts

runter_scroll   movem.l d0/a0-a1,-(sp)   *Register retten
                movea.l logad,a1         *hieraus wird kopiert
                add.l   #31840,a1        *da von unten kopiert wird
                movea.l a1,a0            *hierein wird kopiert
                add.l   #160,a0          *deshalb muß a0 größer sein
                move.w  #7959,d0         *so oft muß kopiert werden
sc_r_1          move.l  -(a1),-(a0)      *kopieren
                dbra    d0,sc_r_1
                move.w  #39,d0           *so oft muß gelöscht werden
sc_r_2          clr.l   -(a0)            *löschen
                dbra    d0,sc_r_2
                movem.l (sp)+,d0/a0-a1
                rts

 END
```

Listing der Grafik-Routinen für den Atari ST (Schluß)

Wegweiser durchs System

Nimmt man die Systemvariablen zu Hilfe kann man auch in Basic tricksen.

Ein wichtiger Teil des Atari ST-Betriebssystems, die Systemvariablen, enthalten Zeiger auf Adressen (Vektoren) oder verschiedene nützliche Speicherstellen.

Der 68000-Prozessor im Atari ST kennt zwei verschiedene Betriebsarten. Einerseits den Supervisormodus, in dem das Betriebssystem arbeitet. Im Usermode laufen dagegen sämtliche Anwenderprogramme, wie Textverarbeitungen, Programmiersprachen und Spiele. Da man in der Regel nur im Supervisormodus auf die Systemvariablen zugreifen darf, muß man vor einem Zugriff vom Usermode in den Supervisormode wechseln. Das funktioniert aber nur in Maschinensprache problemlos. Deshalb bedarf es einiger Vorsicht bei der Verwendung von Systemvariablen in Basic. Das betrifft allerdings nur den Schreibvorgang. Lesen kann man die Systemvariablen immer.

Eine Variable zeigt auf den Anfang des Bildschirmspeichers. Versuchen Sie bitte nicht, ihn zu verschieben, der ST stürzt dann nämlich ab.

Schlüssel zum ST

Von Basic aus kann auf die Variablen mit PEEK und POKE zugegriffen werden. Hexadezimale Zahlen kennzeichnet das vorangestellte »&h«, zum Beispiel »PRINT PEEK (&h44e)«. Die Adressen macht aber in der normalen Schreibweise ein vorangehendes Dollarzeichen ($) kenntlich. Man muß sich aber über folgende Begriffe im klaren sein, um kein Datenchaos zu veranstalten: Ein Bit ist die kleinste Informationseinheit eines Computers. Aus 8 Bits besteht ein Byte, wobei 2 Byte, also 16 Bit, ein Wort (word) ergeben. Zwei Wörter sind ein Doppelwort (long). Da der Speicher des ST wortweise angeordnet ist, stehen in einer Adresse Zahlen bis zu 65535(2^{16}). Zur Umwandlung der einzelnen Worte in Doppelworte (long) oder einzelne Bytes (byte) bedarf es einiger kleiner Tricks.

Zwei Worte in ein Langwort umwandeln: In Adresse $44e steht ein Langwort, das als Zeiger auf den Bildschirmspeicheranfang fungiert. Da diese Adresse jedoch in zwei Worte aufgeteilt ist – der Speicher des ST ist wie gesagt wortweise angeordnet – müssen diese beiden 16-Bit-Zahlen in eine 32-Bit-Adresse konvertiert werden. Dazu benötigt man folgende Formel: Adresse=PEEK(&h44e)*65536+abs (PEEK(&h44e+2)). Diese Konvertierungsart ist bei allen Systemvariablen, die auf eine Adresse zeigen, unumgänglich.

Ein Wort in zwei Byte aufspalten: Bei manchen Systemvariablen darf nur eines der beiden Bytes, aus denen ein Wort besteht, verändert werden. Deshalb muß man das Wort in zwei Byte aufspalten.

Die zwei Byte werden dabei als HI-Byte und LO-Byte bezeichnet, wobei das LO-Byte die Bits 0 bis 7 und das HI-Byte die Bits 8 bis 15 enthält. Ein gutes Beispiel für diese Problematik ist die Systemvariable in Adresse $484. Obwohl es sich hierbei um eine Variable handelt, die nur ein Byte benötigt, ergibt »PRINT PEEK(&h484)« einen 16-Bit-Wert. Das Problem ist, daß man den Wert in $485 nicht ohne weiteres ändern darf. Es ist also nur der Zugriff auf das HI-Byte des Wortes (HI-Byte in $484 ; LO-Byte in $485) erlaubt. Hier hilft die Aufspaltung in HI- und LO-Byte nach folgender Formel:

HI=int(PEEK(&h484)/256):LO=PEEK(&h484)-HI*256.

Normalerweise enthält $484 den Wert 7, in $485 steht eine Null. Beim Aufruf von »PRINT PEEK(&h484)« erhalten wir dennoch den Wert 1792 (7 x 256). Wenn wir nun in $484 eine 5 POKEn wollen, müssen wir die 5 mit 256 multiplizieren und zum LO-Byte addieren, da es sich ja um das HI-Byte handelt. Zum Beispiel:

POKE &h484,LO+5*256

Mit diesem Vorwissen dürfte es kein Problem sein, die Systemvariablen richtig anzuwenden. Zu den einzelnen Variablen sind die Adresse, die Verarbeitungsbreite und der Originalname laut Atari angegeben.

Die wichtigsten Systemvariablen des Atari ST:

$42E long _phystop

Zeiger auf das Ende des RAM-Speichers. Bei einem Computer mit 512KByte enthält diese Adresse den Wert $80000.

$432 long _membot

Zeiger auf die Startadresse des Benutzerspeichers.

$436 long _memtop

Zeiger auf die Endadresse des Benutzerspeichers. Direkt hinter dem Benutzerspeicher liegt dann der Bildschirmspeicher.

$43E word _flock

Ist der Wert dieser Adresse ungleich 0, dann findet gerade ein Diskettenzugriff statt.

$444 word _fverify

Falls diese Speicherstelle einen anderen Wert als 0 enthält, dann wird bei jedem Schreibzugriff auf eine Diskette ein Verify durchgeführt. Das heißt, daß der Atari überprüft, ob das auf die Diskette Geschriebene dem Original entspricht.

$446 word _bootdev

Diese Adresse enthält die Nummer des Diskettenlaufwerks, von dem das Betriebssystem gebootet wurde.

$448 word _palmode

Falls in dieser Speicherstelle ein Wert steht, der ungleich 0 ist, dann befindet sich der ST im PAL-Modus. Normalerweise arbeitet das System im NTSC-Modus. Der PAL-Modus ist die europäische Fernsehnorm und arbeitet mit einem Takt von 50 Hz. Das heißt, der Kathodenstrahl der Bildröhre baut 50mal in der Sekunde ein komplettes (Halb-)Bild auf. In Amerika geschieht dies 60mal in der Sekunde, da die Fernseher dort mit 60 Hz arbeiten. Diese Norm nennt man NTSC-Modus.

$44E long _v_bas_ad

Diese Adresse ist der Zeiger auf den Bildschirmspeicher (SM=screen memory). Bei einem ST mit 512 KByte liegt der SM ab $78000 bei einem 1Mega-ST 512 x 1024 Bytes höher. Wenn man diesen Zeiger verändern will, muß man beachten, daß der Bildschirmspeicher immer an einer 256-Byte-Grenze beginnen muß.

$454 word _nvbls

Länge einer Liste von Adressen, die auf Routinen zeigen, die bei einem VBI abgearbeitet werden.

$456 long _vblqueue

Zeiger auf die Liste der Adressen der Routinen, die bei einem VBI abgearbeitet werden. Wenn eine von diesen Adressen, die in der Liste stehen, in allen 4 Bytes den Wert 0 erhält, kann man an dieser Stelle eine eigene Routine »einschieben«. Die Länge der Liste steht in »_nvbls«.

$45A long _colorptr

Normalerweise steht hier eine Null. Fall sich dieser Wert ändert, wird er als Zeiger auf eine neue Farbpalette interpretiert, die nach dem nächsten VBI geladen wird.

$45E long __screenpt
Ist dieser Wert ungleich 0, wird er nach dem nächsten VBI als neue Bildschirmspeicher-Startadresse interpretiert.
$466 long __frclock
Diese Speicherstelle enthält die Anzahl der ausgeführten VBIs.
$482 word __comload
Ist dieser Wert ungleich 0, dann wurde versucht, das Programm »COMMAND.PRG« zu laden. Falls sich dieses Programm auf der System-Diskette befindet, wird es automatisch gestartet und das GEM nicht geladen.
$484 byte __conterm
Mit den unteren 4 Bits dieses Bytes werden die Attribute für die Konsolen-Ausgabe bestimmt (siehe »Aufspaltung von Worten in HI- und LO-Byte«). $485 darf dabei nicht verändert werden. Die Bits haben folgende Bedeutung:
Bit 0 = 1 : Tastaturklicken einschalten.
Bit 0 = 0 : Tastaturklicken ausschalten.
Bit 1 = 1 : Auto-Repeat der Tasten einschalten.
Bit 1 = 0 : Auto-Repeat der Tasten ausschalten.
Bit 2 = 1 : Glockenton nach Drücken von Controll-G einschalten.
Bit 2 = 0 : Glockenton nach Drücken von Controll-G ausschalten.
Bit 3 = 1 : Bei Aufruf von »conin« wird in den Bits 24 bis 31 »kbshift« zurückgegeben.
Bit 3 = 0 : Keine Rückgabe von »kbshift«.
$486 long __trp14ret
Dieses Langwort enthält die Rücksprungadresse eines TRAP # 14-Aufrufes.
$4A6 word __nflops
Anzahl der angeschlossenen Diskettenlaufwerke.
$4AC word__save__row
Hier wird die Position des Cursors gespeichert, der mit der VT52-Emulator-Funktion »ESC Y« positioniert wird.
$4EE word __dumpflg
Falls die Tasten »Alternate« und »Help« gleichzeitig gedrückt werden, enthält dieses Wort den Wert 0. Im Normalfall wird nach Drücken dieser Tasten die Hardcopy-Routine ausgeführt.
$4F2 long __sysbase
Diese Adresse zeigt auf den Beginn des Betriebssystems.
$4FA long__end__os
Zeiger auf das Ende des Betriebssystems.
$4FE long__exec__os
Zeiger auf den Beginn des AES.

(Sven Krüppel/hb)

Tips für Aufsteiger

Starthilfe für Assembler-Anfänger zum Programmieren mit dem 68000er. Mit diesen Tips finden Sie leichter den Einstieg in die Programmierung dieses leistungsfähigen Prozessors.

Obwohl die Programmierung des 68000er in Assembler sehr komfortabel ist, müssen sich die Assembler-Freaks umstellen. Als kleine Hilfe sind die zwölf Tips gedacht, die verhindern sollen, daß Ihr Computer irgendwo im Nirwana fischt. Das kann durch ein falsch gesetztes Adreßregister schnell geschehen. Im Gegensatz zum 6502, der gerade einen Akku und zwei Zählregister mit je acht Bit aufweisen konnte, bietet der 68000er »nur« acht Datenregister und sieben Adreßregister mit jeweils 32 Bit. Die Datenregister kann man dabei mit dem Akku vergleichen. Die Zählregister (X- und Y-Register) des 6502 sind daher ein Mittelding zwischen Daten- und Adreßregister. Beim Umgang mit diesen Registern, dem Stack und anderem sollen die nun folgenden Tips dienen.

Natürlich ist ein gutes Buch einer der besten Wege, einen neuen Prozessor zu begreifen und leistungsfähige Programme zu schreiben. Meist gehen jedoch diese Bücher nicht auf den Umsteiger ein, dem die Programmierung in Maschinencode zwar geläufig ist, der aber durch die vorherige Arbeit mit einem 8-Bit-Prozessor versucht ist, unwillkürlich unerlaubte Parallelen zu ziehen. Fehler, die viel Zeit kosten und einem den Einstieg schwer machen, lassen sich durch das Beachten von wenigen Hinweisen leicht verhindern.

Beansprucht eine Operation nicht alle 32 Bits eines Datenregisters, werden nur die benötigten niederwertigen Bits geändert. Zum Beispiel

d0=10110100 11101011 10101010 11010101
move.b #ff,d0
→ d0=10110100 11101011 10101010 11111111.

Obwohl der 68000er ein 16-Bit-Prozessor ist, hat jedes Byte eines Wortes eine eigene Adresse.

Zugriffe auf Wörter oder Langwörter dürfen nur von geraden Adressen aus erfolgen. Zum Beispiel 2,4,6,...

Auf Byte zugegriffen werden darf von geraden und ungeraden Adressen.

Der Stack wird beim 68000er vermindert und nicht erhöht. Zum Beispiel Stack :
$ffffffff
$50000000 auf Stack bringen
Stack : $ffffffff
$50000000

Die Adreßregister(A0 bis A6) und der Stack(A7, A7) arbeiten nur mit Wort- oder Langwortoperationen zusammen.

Der 68000er kennt zwei Betriebsarten: Im Supervisormodus läuft das Betriebssystem, im Usermodus sind die Anwenderprogramme aktiv. Beide Betriebsarten haben einen eigenen Stack, wobei jede nur auf ihren Stack zugreifen kann.

Vom Supervisor zum User

Vom Supervisormodus ist ein Wechsel in den Usermode möglich, aber nicht umgekehrt.

Nur im Supervisormodus laufen alle Befehle. Falls man Kommandos, die nur in diesem Modus arbeiten, im Usermode anwendet, führt dies zu einer Ausnahmeverarbeitung (Exception, Fehlerbehandlung).

Das Statusregister besteht aus einem Systembyte und einem Userbyte. Aber nur im Usermode ist der Zugriff auf das Userbyte, in dem zum Beispiel die Flags stehen, erlaubt.

Die Rückkehradresse eines Sprunges (zum Beispiel jmp, jsr) steht immer auf dem Stack.

BCD-Zahlen werden grundsätzlich gepackt verarbeitet, also zwei BCD-Zahlen zu einem Byte zusammengefaßt

(Sven Krüppel/hb)

Tips zum ST-Basic

Hier erwarten Sie wertvolle Hilfen für das Programmieren auf Ihrem Atari ST in Basic.

Da das ST-Basic noch recht wenig Speicherplatz zur Verfügung stellt, muß man sich anders behelfen. Dazu zwei Tips, die zirka 70KByte freigeben.

Benennen sie die beiden Dateien »DESK1.ACC« und »DESK2.ACC« auf der Systemdiskette in »DESK1« und »DESK2« um. Dadurch werden die Unterpunkte des Pull-Down-Menü »DESK« nicht mehr geladen. Man erhält auf diese Weise mehr Speicherplatz. Die Funktionen des Kontrollfeldes, des VT52 Terminal Emulators, der Drucker-Anpassung und der RS232-Einstellung entfallen aber.

Gehen Sie im Basic mit dem Maus-Pfeil auf das Wort »RUN« in der obersten Bildschirmzeile. Nun bewegen Sie den Pfeil auf Menüpunkt »Buf Graphics« und drücken einmal auf die linke Maustaste, falls links von »Buf Graphics« ein Haken sichtbar ist, damit dieser verschwindet. Wenn dieser Menüpunkt aktiviert ist (Haken auf der linken Seite), wird die Grafik, die sich momentan im Output-Fenster befindet, beim Öffnen eines anderen Fensters abgespeichert. Da der Bildschirmspeicher 32KByte lang ist, steht Ihnen dieser Platz zur freien Verfügung, wenn dieser Punkt nicht angeklickt ist. Übrigens dürfen Sie »Buf Graphics« nur anklicken, wenn kein Programm im Arbeitsspeicher liegt. Sonst würde es gelöscht.

Zahlenspielereien

Das ST-Basic kann mit Dezimal-, Oktal- und Hexadezimalzahlen arbeiten. Um zum Beispiel Adressen in Hexzahlen anzugeben, muß man vor die Zahl »&h« setzen. Zum Beispiel &hFF, &hA0. Zahlenumwandlungen funktionieren folgendermaßen: Dezimal-Hexadezimal : »? HEX$(Dez-Zahl)«,
zum Beispiel »? HEX$(24)«
Dezimal-Oktal: »? OCT$(Dez-Zahl),«
zum Beispiel »? OCT$(24)«
Hexadezimal-Dezimal: »? &h(Hex-Zahl)«,
zum Beispiel »? &hFA
Oktal-Dezimal: »? &o(Oktal-Zahl)«,
zum Beispiel »? &o66«.

Weiße Schrift auf schwarzem Hintergrund erhält man beim Monochrom-Monitor, indem man im Kontrollfeld die drei Regler auf der linken Seite nach unten schiebt.

Vor Bildschirmausgaben, die auf VDI-Routinen oder Bildschirmspeicher beruhen, sollte man den Mauscursor immer unsichtbar machen. Der Bildschirminhalt kann verfälscht werden, wenn die Maus auf einer Stelle steht, an der etwas geändert wird. Im Basic muß ein String, der über eine VDI-Funktion ausgegeben wird, nicht mit ASCII 0 beendet werden, wie in C.

Der gesamte Zeichensatz ist im englischen Logo-Handbuch auf den Seiten 63 bis 65 abgedruckt. Falls Sie ein Zeichen benötigen, das nicht durch Tastenkombinationen erhältlich ist, können Sie dieses Zeichen mit dem Befehl »? chr$(zeichencode)« ausgeben. Zum Beispiel schreibt »? chr$(189)« ein Copyright-Zeichen.

Der normale PRINT-Befehl verarbeitet die deutschen Sonderzeichen leider nicht. Sollten Sie sie dennoch benötigen, können die Zeichen entweder mit »? chr$(zeichencode)« oder über die Text-Funktionen des VDI ausgegeben werden.

(Sven Krüppel/hb)

Allerlei für Jedermann

Unter diesen Tips ist für jeden etwas dabei, egal ob Anfänger oder Profi.

Wer beim Atari ST in Basic professionell wirkende Programme schreiben will, muß schon auf einige Tricks zurückgreifen. Wenn Sie die angeführten Codes gezielt einsetzen, schreiben Sie Basic-Programme, die sich äußerlich nicht von TOS-Anwendungen unterscheiden. Aber auch für kleinere Aktionen, zum Beispiel Bildschirm löschen, können diese Codes sehr nützlich sein.

Den VT52-Emulator sollte jeder ST-Anwender kennen, da er den Weg in die vielen Mailboxen öffnet. Man kann die empfangenen Daten zwar nicht abspeichern, aber es reicht, um mal ein bißchen DFÜ-Luft zu schnuppern. Uns interessieren hier aber nicht die Datenfernübertragungen, sondern die Steuercodes, die der VT52-Emulator zur Verfügung stellt.

Im ST-Basic gibt es die beiden Befehle »INP« und »OUT«. Mit diesen Funktionen kann man Zeichen einlesen und ausgeben. Dabei gilt folgendes Befehlsformat: a=inp(b) ; Out b,a

»a« ist ein beliebiger ASCII-Code. Weist man a den Wert 65 zu, dann wird durch »Out 2,a« ein »A« auf dem Bildschirm ausgegeben. Gibt man »a=inp(2)« ein, erhält man in der Variable a nach Drücken der Taste »A« eine 65

»b« ist die Kennummer des angesprochenen Geräts. Folgende Gerätekennummern sind vorhanden :

0 : Drucker (Centronics-Schnittstelle)
1 : Modem (V.24 Schnittstelle)
2 : Tastatur und Bildschirm (Konsole)
3 : Midi-Port
4 : Tastatur-Prozessor

Vom Basic läßt sich der Tastatur-Prozessor leider nicht vernünftig ansteuern, da es zu langsam ist. Bei den Tests haben wir jeweils die Konsole (2) angesprochen.

Kleiner Code, große Wirkung

Nun zu den eigentlichen Codes: Es gelten in erster Linie die normalen ASCII-Codes. Die Codes von 0 bis 31, die bei manchen Computern zur Drucker- und Bildschirmsteuerung benutzt werden, sind aber beim ST nicht alle vorhanden. Von diesen Codes existieren nur folgende :

Out 2,7: läßt die wohlbekannte Glocke erklingen.

Out 2,8: löscht das Zeichen links vom Cursor. Der Cursor steht normalerweise in der linken oberen Ecke. Im Basic und in den meisten GEM-Programmen ist er unsichtbar, läßt sich aber anschalten. Diesen Cursor benutzen auch TOS-Anwendungen.

Out 2,9: Tabulator.

Out 2,10: Linefeed (LF, Zeilenvorschub).

Out 2,11: Vertikal Tabulator (VT), löst aber einen Zeilenvorschub aus.

Out 2,12: Form Feed (FF), auch Zeilenvorschub.

Out 2,13: Carriage Return (CR, Sprung an den Zeilenanfang).

Out 2,27: Escape (ESC).

Die restlichen Steuercodes benötigen zwei oder mehr Parameter. Der erste ist immer Escape (27).

Out 2,27, Out 2,65: Cursor hoch, der Cursor wird eine Zeile nach oben geschoben. Falls er sich schon in der ersten

Zeile befindet, zeigt dieser Code keine Wirkung.

Out 2,27, Out 2,66: Cursor runter. Dieser Code positioniert den Cursor eine Zeile tiefer. Falls er schon in der untersten Bildschirmzeile ist, geschieht nichts.

Out 2,27, Out 2,67: Cursor rechts. Der Cursor wird ein Zeichen nach rechts verschoben. Am Zeilenende bleibt er stehen.

Out 2,27, Out 2,68: Cursor links. Der Cursor wird eine Position nach links verschoben (Backspace).

Out 2,27, Out 2,69: Bildschirm löschen. Der Bildschirm wird vollständig gelöscht und der Cursor springt in die obere linke Ecke zurück.

Out 2,27, Out 2,72: Cursor Home. Der Cursor springt in die obere linke Ecke zurück, ohne den Bildschirm zu löschen.

Out 2,27, Out 2,73: Cursor hoch mit Scrolling wie »Cursor hoch«. Allerdings wird der Bildschirminhalt nach unten verschoben (gescrollt), falls sich der Cursor in der obersten Zeile befindet. In der ersten Zeile wird dann eine Leerzeile eingefügt.

Out 2,27, Out 2,74: unteren Teilbildschirm löschen. Mit dieser Sequenz kann der Bildschirm unterhalb der Cursorposition gelöscht werden.

Out 2,27, Out 2,75: Bereich bis zum Zeilenende löschen. Der Zeileninhalt, der sich rechts vom Cursor befindet, wird gelöscht.

Out 2,27, Out 2,76: Zeile einfügen. An der momentane Cursorposition wird eine Leerzeile eingefügt und der Bildschirminhalt eine Zeile nach unten gescrollt.

Out 2,27, Out 2,77: Zeile löschen. Die Zeile, in der sich der Cursor gerade befindet, wird gelöscht und der Bildschirminhalt eine Zeile nach oben gescrollt. Die unterste Bildschirmzeile ist danach leer. Der Cursor steht anschließend in der ersten Spalte der Zeile unter der gelöschten Zeile.

Out 2,27, Out 2,89, Out 2,32+x, Out 2,32+y: Cursor positionieren.

x von 0 bis 79; y von 0 bis 24.

Der Cursor kann mit Hilfe dieser Funktion beliebig auf dem Bildschirm positioniert werden.

Out 2,27, Out 2,98, Out 2,f: Schriftfarbe verändern. Die Schrift kann bei angeschlossenem Farbmonitor bis zu 16 verschiedene Farben annehmen. Die Variable f enthält dabei die Farbnummer. Bei einem Monochrom-Monitor stehen nur die Farben 0 (weiß) und 1(schwarz) zur Verfügung.

Die auf diese Weise eingestellte Farbe ist nur bei Zeichen wirksam, die dem Out-Befehl folgen. Die Print-Anweisung ignoriert diese Funktion.

Out 2,27, Out 2,99, Out 2,f: Hintergrundfarbe wählen. Diese Funktion ist mit der vorhergegangenen identisch, nur daß hier keine Veränderung der Schrift-, sondern der Hintergrundfarbe eintritt.

Out 2,27, Out 2,100: Oberen Teilbildschirm löschen. Diese Funktion löscht den Bildschirm von der ersten Zeile bis zu der Zeile, in der sich der Cursor momentan befindet.

Out 2,27, Out 2,101: Cursor sichtbar machen. Der sonst unsichtbare Cursor wird durch diese Funktion sichtbar gemacht.

Out 2,27, Out 2,102: Cursor unsichtbar machen. Diese Funktion schaltet die Cursordarstellung ab. (Normalzustand)

Out 2,27, Out 2,106: Cursorposition speichern. Die momentane Cursorposition wird zwischengespeichert.

Out 2,27, Out 2,107: Cursor auf alte Position setzen. Der Cursor kann hiermit auf die mit »Cursor speichern« bestimmte Position gesetzt werden. Der Aufruf dieser Funktion löscht die Zeile, in der sich der Cursor gerade befindet. Dabei wird kein Scrolling durchgeführt.

Out 2,27, Out 2,111: Bereich bis zum Zeilenanfang löschen. Der Zeileninhalt vom ersten Zeichen bis zur momentanen Cursorposition wird gelöscht.

Out 2,27, Out 2,112: Inverse Schrift. Sämtliche Zeichen, die mit dem Out-Befehl nach Aufruf dieser Funktion ausgegeben werden, erscheinen in inverser Schrift. Bei inverser Schrift, wird das, was bei der normalen Schrift weiß ist, schwarz und umgekehrt.

Out 2,27, Out 2,113: Normalschrift. Diese Funktion schaltet die vorhergegangene aus.

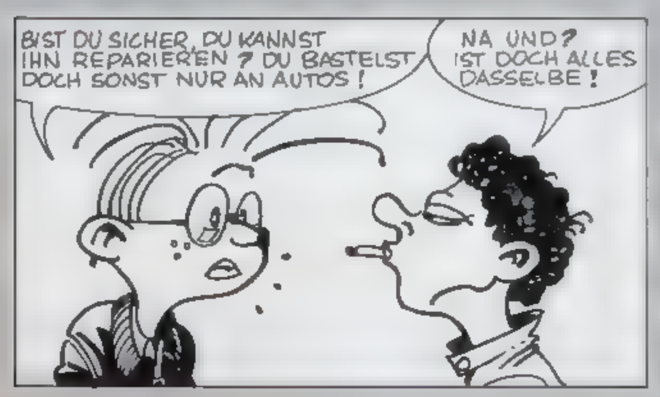
Out 2,27, Out 2,118: Überlauf ein. Durch Aufruf dieser Funktion springt der Cursor nach Erreichen der letzten Spalte in die erste Spalte der folgenden Zeile.

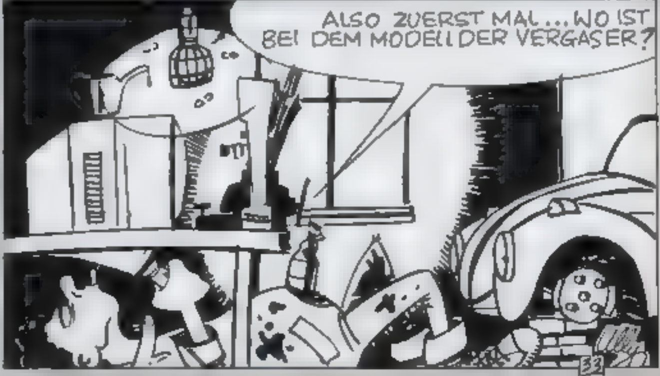
Out 2,27, Out 2,119: Überlauf aus. Der Cursor springt nach Aufruf dieser Funktion nicht mehr an den Anfang der nächsten Zeile, sondern alle folgenden Zeichen werden in die letzte Spalte geschrieben.

Setzen Sie diese Funktionen gezielt ein, dann können Sie Ihre Programme sehr leicht perfektionieren. Wer das OUTPUT-Fenster nicht benötigt, kann den Bildschirm mit »Out 2,27, Out 2,69« löschen und anschließend mit VDI-Funktionen oder mit dem Out-Befehl beschriften. Die Umrandung des OUTPUT-Fensters erscheint jedoch wieder, wenn man einen PRINT- oder INPUT-Befehl verwendet.

(Sven Krüppel/hb)

kosinus von GUBA & ULLY





GEM-Komfort mit Basic

Auch in selbstgeschriebenen Basic-Programmen braucht man auf GEM-Unterstützung nicht zu verzichten.

Das Grundgerüst des »Maus-Editors« ist das ST-Basic. Da man mit dieser Sprache bei der GEM-Programmierung aber nicht sehr weit kommt, muß man den ST schon genauer kennen, um ein solches Bildschirmlayout zu programmieren. Im wesentlichen handelt es sich dabei um die Benutzung von GEM-Routinen in Basic und den Bildschirmspeicher.

Wenn man vom GEM spricht, muß man zwischen dem VDI (Virtual Device Interface) und dem AES (Application Environment System) unterscheiden.

Das VDI stellt dem Programmierer grundsätzliche Grafikfunktionen zur Verfügung, wogegen das AES mehr für die Verwaltung der Windows und Drop-Down-Menüs zuständig ist. Da das Programmieren mit AES-Routinen von Basic aus mehr als schwer und teilweise unmöglich ist, wollen wir uns nur mit dem VDI befassen

Man kann die VDI-Routinen als Unterprogramme verstehen, die vom Basic angesprungen werden. Der Sprung erfolgt dabei durch den Befehl »vdisys«. Vorher benötigt das VDI aber verschiedene Parameter, zum Beispiel die Funktionsnummer, damit es feststellen kann, welche Routine angesprochen ist.

Wenn Sie sich die Zeile 10060 des »Maus-Editors« ansehen, werden Sie feststellen, daß vor den »vdisys«-Befehl mehrere Werte gePOKEd wurden. Diese Werte übernimmt das VDI in fünf festgelegten Variablen, wobei jede auf einen Speicherbereich zeigt, Array genannt. Dies geschieht, weil die Parameter in anderen Sprachen auch meist in Arrays abgelegt werden. Die fünf Speicherbereiche haben alle eine ganz spezielle Aufgabe. Im contrl-Array übergibt der Programmierer dem VDI sämtliche Control-Parameter, die Nummer des angesprochenen Geräts, die Funktionsnummer (Opcode) und die Länge der anderen Arrays.

Das VDI erwartet vom Basic im contrl-Array folgende Parameter:

contrl:	Opcode (Funktionsnummer)
contrl+2:	Länge des ptsin-Arrays
contrl+6:	Länge des intin-Arrays
contrl+10:	eventuell Unterfunktionsnummer

In contrl+4 und contrl+8 übergibt das VDI nach dem Aufruf dann die Länge des ptsout- und intout-Arrays. Bei unserem ersten Beispiel ist nur das contrl-Array von Interesse, da andere von dieser Funktion keine Parameter erwarten.

Ein Tastendruck schon macht in Basic den Maus-Cursor automatisch unsichtbar. Diese Funktion findet auch beim »Maus-Editor« Verwendung, da dieser bei verschiedenen Grafikfunktionen sehr stören kann. Unser erstes Demo schaltet also den Mauskursor aus. Diese Routine steht in den Zeilen 10080 bis 10110. Doch erst zu einer Besonderheit des ST-Basic. Es gestattet dem Programmierer, Unterprogramme über Labels anzuspringen. Unter einem Label versteht man eine Art Variable, die die Startadresse einer Routine (Unterprogramm) enthält. Wenn unser Programm nun in die Routine zum Ausschalten des Maus-Cursors verzweigen soll, brauchen wir nur noch »Gosub hidecursor« an Stelle von »gosub 10080« zu schreiben. Allerdings muß diese Adresse in der ersten Zeile der Subroutine stehen, gefolgt von einem Doppelpunkt (Zeile 10090).

Da in diesem Fall die VDI-Routine nur Parameter im contrl-Array erwartet, übergeben wir in Zeile 10100 die benötigten Werte. Die Funktion erwartet von uns in contrl den Opcode (POKE contrl 123) und die Länge der anderen Arrays. Da diese Arrays keine Parameter enthalten, wird in contrl+2 und contrl+6 eine 0 abgelegt (POKE contrl+2,0:POKE contrl+6,0). Am Schluß ruft man die Funktion mit »vdisys« auf. Da das VDI aber vom Basic die Parameter in bestimmten Speicherstellen erwartet und der Speicher wortweise angeordnet ist, verschieben sich die Adressen der Parameter Es müssen also alle solchen Adressen, die diesem Artikel entnommen und in Basic angewendet werden sollen, mit 2 multipliziert werden. Dies wird deutlicher, wenn wir die komplizierteren VDI-Funktionen besprechen.

Um den Maus-Cursor nun wieder einzuschalten, bedarf es einer anderen Funktion. Diese benötigt aber auch einen Parameter im intin-Array; contrl+6 muß also 1 sein (POKE contrl+6,1).

Im intin-array werden vom VDI meist die Werte verlangt, die zur Abarbeitung fast aller Funktionen von Nöten sind. Dabei kann es sich um Texte, Farbwerte oder ähnliches handeln. Der Wert, der das Einschalten des Maus-Cursors bewirkt, enthält die Information, ob die Anzahl der hidecursor-Aufrufe ignoriert werden soll oder nicht. Der ST zählt nun intern die hidecursor-Aufrufe. Übergibt der Programmierer im intin-Array nun den Wert 0, wird der Maus-Cursor auf jeden Fall aktiviert. Registriert er indessen eine 1, wird der hidecursor-Zähler um 1 erniedrigt.

Beim »Maus-Editor« ist der Cursor aber auf jeden Fall erforderlich. Wir poken also eine 0 in das intin-Array. (Zeile 10040 bis 10060). Die restlichen Parameter geben die Länge des ptsin-Arrays (0) und den Opcode (122) an.

So präsentiert sich der »Maus-Editor« im GEM-Look. Trotz Basic stellt das Programm alle Funktionen der Benutzeroberfläche zur Verfügung.

Eine andere interessante Routine verändert die Schriftart. Diese Routine liegt in den Zeilen 10260 bis 10290, wobei der intin-Array eine Angabe der Schriftart erwartet. Das GEM stellt 33 verschiedene Varianten zur Verfügung, die sich aus Kombinationen der einzelnen Grundstile zusammensetzen.

Wertigkeit	Stil
0	normal
1	fett
2	hell
4	kursiv
8	unterstrichen
16	auseinandergezogen
32	schattiert

Fette und auseinandergezogene Kursivschrift erfordert den Wert 21 (1+4+16) im intin-Array. Die restlichen Parameter entnehmen Sie bitte der Programmübersicht.

Diese Routine arbeitet leider nicht einwandfrei mit dem Print-Befehl des ST-Basic zusammen, weil sie bei einer Print-Anweisung von einer Schriftbreite von 8 Pixel ausgeht. Viele Schriftarten erscheinen aber in breiteren Zeichen, die dann teilweise überschrieben werden. Um dies zu vermeiden, müssen Sie einen Text über die entsprechende VDI-Funktion ausgeben, die später beschrieben wird.

Funktionen, die die Schrift in irgendeiner Weise verändern, muß man vor dem Verlassen des Hauptprogramms mit den ursprünglichen Parametern noch einmal aufrufen.

Die Routine zum Verändern der Schriftgröße läuft ähnlich der oben stehenden ab.

Im ptsin-Array erhält der VDI immer irgendwelche Bildschirmkoordinaten, die paarweise behandelt werden. Als Länge für das ptsin-Array muß er in contrl+2 die Anzahl der Koordinatenpaare übergeben, die er zur Ausführung der Routine benötigt, Informationen über die Ausgabeposition des Textes zum Beispiel.

Textvariationen

Die Routinen zum Zeichnen eines ausgefüllten Rechtecks (Zeile 10120 – 10170) zeigen dies sehr deutlich. Im ptsin-Array werden vom VDI die Koordinaten der beiden diagonal gegenüberliegenden Eckpunkte des Rechtecks erwartet. Da es sich um zwei Bildschirmkoordinaten (mit je zwei Parametern) handelt, gibt man als Länge für diese Array in contrl+2 den Wert 2 an.

Grafikoperationen können über das VDI den gesamten Bildschirm ansprechen. Man muß sich also nicht an die Bildschirmbegrenzungen halten, die für Basic gelten. Die Füllfarbe bestimmt ausnahmsweise der Basic-Befehl »color«, da sich ein Einsatz des VDI hier nicht lohnt. Diese Routine benötigt zusätzlich zum Opcode eine Unterfunktionsnummer, die man in contrl+10 ablegt. Da eine ganze Reihe von Grafikfunktionen über den Opcode 11 angesprochen werden, unterscheiden sie sich lediglich durch ihre Unterfunktionsnummer

GEM eignet sich sehr gut zur Textausgabe. Sie können den Text links und/oder rechts justiert oder auch zentriert editieren. Es ist klar, daß dies etwas aufwendiger ist als über »Print«, aber die Mühe lohnt sich. Das entsprechende Unterprogramm finden Sie in den Zeilen 10180 bis 10250. Unter Wortdehnung versteht man Abstände zwischen den einzelnen Worten, unter Zeichendehnung die einzelnen Zeichenabstände. Wenn einer dieser beiden Parameter den Wert 1 enthält, muß man die Länge der Ausdehnung angeben. Auf diese Weise kann ein Text die ganze Bildschirmbreite beanspruchen. Der Text muß dann in ASCII-Code im intin-Array übergeben werden.

Das intout- und ptsout-Array benötigt man bei der Funktion mauspos (Zeile 10000 bis 10030). Das VDI erwartet nur den Opcode und die Länge der Eingabe-Arrays, gibt aber in contrl+2 die Anzahl der Koordinatenpaare, im ptsout- und in contrl+8 die Länge des intout-Arrays zurück.

Die beiden Parameter im ptsout-Array enthalten die x- und y-Position des Maus-Cursors und intout beinhaltet den Status der Maustaste (gedrückt oder nicht gedrückt).

Diese Funktion macht ersichtlich, wie langsam das Basic ist. Wenn Sie die Maus sehr schnell bewegen, reicht die Geschwindigkeit lange nicht aus, um die Variablen xmaus und ymaus zu aktualisieren. So kann man mit der Maus ein Drop-Down-Menü »herunterziehen«, ohne daß vorher die Alarmglocke ertönt.

Prinzipiell wissen Sie nun alles, um in Basic mit VDI-Routinen zu arbeiten. Da es aber weit über 100 verschiedene VDI-Funktionen gibt, möchte ich auf »Das große GEM-Buch zum Atari ST« verweisen.

Bit-Spielerei

In den Zeilen 260 und 270 wird der Bildschirmspeicher mit dem Befehl »not« für den oberen und unteren Rand invertiert. Alles, was vorher weiß war, wird schwarz und umgekehrt. Da auch der Speicher des ST wortweise angeordnet ist, braucht nur jedes zweite Byte verändert werden (step 2). Dieser Vorgang ist leicht verständlich, weil da, wo ein Bit im Bildschirmspeicher gesetzt war, dieses Bit gelöscht wurde. Will man aber freier mit dem Bildschirmspeicher umgehen, also gezielt Punkte über den Bildschirmspeicher setzen, wird es ein wenig komplizierter, da der ST mit vorzeichenbehafteten Binärzahlen arbeitet. Normalerweise erhält man den Wert 65535, wenn alle 16 Bits eines Wortes gesetzt sind. Da der ST aber mit einem Vorzeichen-Bit arbeitet, erhält man nach »POKE&h78000,65535« in $78000 den Wert –1.

Das jeweils höchstwertigste Bit, Bit 15 bei einem Wort, ist das Vorzeichenbit. Bei gesetztem Bit wird die Zahl negativ, andernfalls positiv. Auf diese Weise sind vorzeichenbehaftete Zahlen von –32768 bis 32767 darstellbar. Wenn man nun alle Bits setzt, gilt das auch für das Vorzeichenbit. Die Zahl ist negativ. Das wäre also die Erklärung, warum aus »Peek(&h78000)« eine negative Zahl resultiert. Aber wie kommt der ST dann auf die 1?

Bei der Zahl –1 handelt es sich um das Zweierkomplement von 65535. Das Einerkomplement einer Zahl kennen Sie schon, dabei werden alle Einsen in Nullen gewandelt und umgekehrt. Das Einerkomplement einer Zahl erhält man also durch den Befehl »not«. Für das Zweierkomplement muß man erst das Einerkomplement ermitteln und den Wert 1 dazu addieren. Das folgende Beispiel verdeutlicht:

65525	= 1111 1111 1111 1111
Einerkomplement	= 0000 0000 0000 0000
Zweierkomplement	= 0000 0000 0000 0001

Wandelt man diese Zahl nun in eine Dezimalzahl um, resultiert der Wert 1. Das Minus vom gesetzten Vorzeichenbit davorgestellt, ergibt die Zahl –1. Es stimmt also. Sie brauchen keine Angst zu haben, daß Sie jetzt ständig rechnen müssen, nur um einen Wert in irgendeine Speicherstelle zu POKEn. Der ST nimmt die Zahlen ohne Vorzeichen auch an. Diese Verfahren müssen Sie aber anwenden, wenn der Computer einen negativen Wert ausgibt. Das vorangegangene Beispiel wollen wir nun einmal in umgekehrter Reihenfolge betrachten.

Gegeben: –1

1 (–1 ohne Vorz.)	= 0000 0000 0000 0001
Einerkomplement	= 1111 1111 1111 1110
Zweierkomplement	= 1111 1111 1111 1111

Nun sollten Sie in der Lage sein, den »Maus-Editor« voll zu verstehen.

Mit Hilfe dieses Programmes können Sie den Maus-Cursor, der nomalerweise in Basic als Pfeil dargestellt wird, komfortabel editieren. Er ist dann in der Lage, ein ferti-

ges Basic-Programm, namens »CURSOR.BAS«, auf Diskette zu speichern. Ein Maus-Cursor setzt sich immer aus drei Teilen zusammen:

Die Cursorform (das, was Sie sehen, wenn der Maus-Cursor über eine weiße Fläche bewegt wird), die normalerweise schwarz ist.

Die Cursormaske, die weiße Umrandung der Cursorform, die nur dann sichtbar ist, wenn die Cursorform sich nicht vom Hintergrund abhebt.

Der Aktionspunkt, der Punkt der Maus, der die exakte Position der Maus kennzeichnet.

Wenn man nun einen neuen Maus-Cursor erstellen will, sollte man die Menüpunkte auf der rechten Bildschirmseite von oben nach unten ausfuhren.

Die Auswahl der Menüpunkte erfolgt, indem man den Maus-Cursor auf den gewünschten Menüpunkt steuert und die Maustaste drückt. Sofort färbt sich das ausgewählte Feld schwarz und der Menüpunkt ist aktiviert.

Nun zu den Menüpunkten im Einzelnen:

Cursorform editieren: Wenn sie diesen Menüpunkt anwählen, können Sie die Cursorform editieren. Dies sollte immer der erste Schritt beim Basteln eines neuen Maus-Cursors sein.

Auf der linken Bildschirmseite sehen Sie ein großes Quadrat, das horizontal und vertikal in 16 Felder aufgeteilt ist. Jedes dieser einzelnen Felder stellt einen Punkt der späteren Cursorform dar. Weiße Felder bedeuten, daß dieser Punkt nicht sichtbar ist, schwarze Felder zeigen, daß der gewünschte Punkt beim späteren Maus-Cursor sichtbar, also schwarz, ist. In dem großen weißen Quadrat unten rechts sehen Sie die Cursorform in Originalgröße. Um nun einen Punkt der Cursorform festzulegen, gehen Sie mit dem Mauskursor auf das gewunschte Feld und drucken die linke Maustaste. Ein weiteres Drucken der Taste löscht den Punkt wieder und das Feld färbt sich schwarz. Falls Sie einen Punkt der Cursorform an eine Stelle legen, an der schon ein Punkt der Cursormaske liegt, wird der Punkt der Cursormaske gelöscht und überschrieben.

Bedienungsanleitung

Cursormaske editieren: Die Bedienung dieses Menüpunktes ist dem vorherigen sehr ähnlich, allerdings werden Punkte der Cursormaske nicht schwarz, sondern gepunktet dargestellt. Dabei hat das Setzen eines Punktes auf einem Punkt der Cursorform keine Auswirkungen. Die Originalgröße der Cursormaske sehen Sie im schwarzen Quadrat unten rechts.

Aktionspunkt festlegen: Die Position des Aktionspunktes kennzeichnet ein diagonales Kreuz. Der Aktionspunkt kann dabei irgendwo in dem 16 x 16 Punkte-Feld liegen Es ist vollkommen egal, ob die gewunschte Position bereits belegt ist.

Cursor aktivieren Nachdem Sie die ersten drei Schritte befolgt haben, können Sie den neu definierten Cursor aktivieren.

Achtung: Sie dürfen die Maus erst wieder bewegen, wenn das Menüfeld wieder weiß geworden ist!

Programm speichern: Damit Sie in Ihren Basic-Programmen profitieren, wird ein komplettes Basic-Programm namens »CURSOR.BAS«, auf Diskette abgelegt. Dieses Programm beginnt immer ab Zeile 12000 und läßt sich direkt mit »run« starten.

Achtung: Sie dürfen die Maus erst wieder bewegen, wenn das Menüfeld wieder weiß geworden ist!

Programm beenden: Wenn Sie den Maus-Editor verlassen wollen, sollten Sie dies nur über diesen Menüpunkt tun. Falls Sie dennoch mit »Contrl+c« abbrechen, entfallen Tastatur-Klick und Auto-Repeat. Das kann anderen Programmen schaden.

(Sven Krüppel/hb)

Programmübersicht

Zeile	Wirkung	
110 - 145	Initialisierung	
150 - 270	Bildschirmrand beschriften	
280 - 385	Menu zeichnen	
390 - 410	Bit-Feld zeichnen	
450 - 640	Mausabfrage und Verzweigung in Subroutines	
650 - 730	Cursorform	
740 - 830	Cursormaske	
840 - 960	Aktionspunkt	
970 - 1040	Cursor aktivieren	
1050 - 1220	Speicherroutine	
10000 - 10030	Abfrage der Mausposition und Maustasten Status	
	Opcode	124
(mauspos)	Länge ptsin und intin	0
	x-Position der Maus	peek(ptsout)
	y-Position der Maus	peek(ptsout+2)
	Maustasten-Status	peek(intout)
10040 - 10070	Maus Cursor einschalten	
	Opcode	122
(showcursor)	Länge ptsin	0
	Länge intin	1
	Wert in intin	0 oder 1
	Ist der Wert im intin-Array gleich 0 dann spielt es keine Rolle wie oft hidecursor aufgerufen wurde Bei 1 wird der Cursor nur eingeschaltet, wenn die Anzahl der hidecursor und showcursor-Aufrufe übereinstimmt	
10080 - 10110	Maus-Cursor ausschalten	
	Opcode	123
(hidecursor)	Länge ptsin und intin	0
	siehe oben	
10120 - 10170	Rechteck zeichnen	
	Opcode	11
(rechteck)	Unterfunktionsnummer	1
	Länge ptsin	2
	Länge intin	0
	ptsin	1 x-Koordinate
	ptsin+2	1 y-Koordinate
	ptsin+4	2 x-Koordinate
	ptsin+6	2. y-Koordinate
10180 - 10250	Text ausgeben	
	Opcode	11
(text)	Unterfunktionsnummer	10
	Länge ptsin	2
	Länge intin	len(t$)+2
	ptsin	x Position des Textes
	ptsin+2	y-Position des Textes
	ptsin+4	Textdehnung
	intin	Wortdehnung
	intin+2	Zeichendehnung
	intin+4 bis intin+n*2	Text in ASCII
10260 - 10290	Schriftart andern	
	Opcode	106
(effekt)	Länge ptsin	0
	Länge intin	1
	intin	Schriftart
	Schriftarten 0 : normal 1 : fett 2 : hell 4 : kursiv 8 : unterstrichen 16 auseinandergezogen 32 schattiert beliebige Kombinationen erreicht man durch Addition der Wertigkeiten	
10300 - 10330	Textgröße verändern	
	Opcode	107
(größe)	Länge ptsin	0
	Länge intin	1
	intin	Texthöhe
	Texthöhen. 1,9,10,16,18,20	
10340 - 10420	Auswertung	
(auswertung)	Umwandlung der Bit-Muster in Datenworte	

```
10 rem **********************************
20 rem * (c) 1985  SVK-Soft             *
30 rem *                                *
40 rem * written by  Sven Krüppel       *
50 rem *             Linderner Str.29   *
60 rem *             5138 Heinsberg     *
70 rem *                                *
80 rem * exclusive for 'Happy Computer' *
90 rem *                                *
100 rem *********************************
110 fullw 2:clearw 2:out 2,27:out 2,69:
color 1,1,1:gosub hidecursor
120 dim pixel(16,16),t$(6),maske(16,16),
mausw(16),maskw(16)
530 for x=1 to 16:for y=1 to 16:
pixel(x,y)=0:maske(x,y)=0:next:next
140 a=0:z=1:ax=1:ay=1:wdehn=0:zdehn=0:
sm=peek(&h44e)*65536+abs(peek(&h450))
145 hi=int(peek(&h484)/256):lo=peek
(&h484)-256*hi:poke &h484,4*256+lo
150 rem ###### Titel ################
160 s=5:gosub effekt:hoehe=160:gosub
groesse
170 x=0:y=33:t$(1)=" SVK-SOFT":gosub
text:hoehe=9:gosub groesse
180 x=155:y=14:t$(1)="prsentiert":gosub
text:hoehe=20:gosub groesse
190 s=25:gosub effekt:wdehn=0:zdehn=1:
xdehn=320:x=280:y=33:t$(1)="Maus -
Editor"
200 gosub text:zdehn=0:wdehn=0
210 s=4:gosub effekt:hoehe=1:gosub
groesse
220 x=0:y=378:t$(1)=" (c) 1985 by":gosub
text
230 s=16:gosub effekt:hoehe=10:gosub
groesse:wdehn=1:zdehn=1:xdehn=630
240 x=0:y=395:t$(1)=" Sven Krüppel,
Linderner Str.29, 5138 Heinsberg"
250 gosub text
260 for i=sm to sm+3438 step 2:poke
i,not peek(i):next
270 for i=sm+29760 to sm+32000 step
2:poke i,not peek(i):next
280 rem ###### Menü #################
290 x=18:s=10:for y=1 to 11 step 2:color
1,1,1,1,0:gosub rechteck:next
300 x=19:s=30:y=15:gosub rechteck
310 x=23:color 1,1,1,1,1:gosub rechteck
320 t$(1)="Cursorform editieren"
330 t$(2)="Cursor-Maske editieren"
340 t$(3)="Aktionspunkt festlegen"
350 t$(4)="Cursor aktivieren"
360 t$(5)="als Programm speichern"
370 t$(6)="Programm beenden"
380 xdehn=220:x=380:for z=1 to 6:y=z*
40+26:gosub text:next:wdehn=0:zdehn=0
385 s=0:gosub effekt:s=10
390 rem  ##### Feld zeichnen ##########
400 color 1,1,1
410 for i=10 to 330 step 20:linef 10,i,
330,i:linef i,10,i,330:next
420 rem
430 rem        HAUPTPROGRAMM
440 rem
450 goto 940
460 gosub showcursor
470 gosub mauspos
480 if xmaus<10 or xmaus>370 or ymaus<10
or ymaus>330 then out 2,7:goto 460
490 if button=0 then 460
500 gosub hidecursor
510 x=cint(xmaus/20):y=cint(ymaus/20):if
x<>18 or y>11 then 600
520 if a=0 then 540
530 hilf=y:y=a:color 1,1,0,0,0:gosub
rechteck:y=hilf
540 a=y
550 color 1,1,1,1,1:gosub rechteck
560 if a=7 then 980
570 if a=9 then 1060
580 if a=11 then gosub showcursor:poke
&h484,7*256+lo:end
590 goto 460
600 if x<1 or x>16 or y<1 or y>16 then
[illegible]
610 if a=1 then 660
620 if a=3 then 750
630 if a=5 then 850
640 goto 460
650 rem ###### Cursorform ##########
660 sprung=0:if maske(x,y)=1 then
sprung=1:goto 770
670 if pixel(x,y)=0 then color
1,1,1,1,1:gosub rechteck
680 if pixel(x,y)=0 then color
1,1,1:linef 373+x,291+y,373+x,291+y:goto
710
690 if pixel(x,y)=1 then color
1,1,0,0,0:gosub rechteck
700 if pixel(x,y)=1 then color
1,1,0:linef 373+x,291+y,373+x,291+y
710 pixel(x,y)=1-pixel(x,y)
720 if ax=x and ay=y then 910
730 goto 460
740 rem ##### Cursormaske ##########
750 if pixel(x,y)=1 then 460
770 if maske(x,y)=0 then color
1,1,1,0,2:gosub rechteck
780 if maske(x,y)=1 then color
1,1,1,0,0:gosub rechteck
790 if maske(x,y)=0 then color
1,0,0:linef 453+x,291+y,453+x,291+y
800 if maske(x,y)=1 then color
1,0,1:linef 453+x,291+y,453+x,291+y
810 maske(x,y)=1-maske(x,y)
820 if sprung=1 then 670
830 if x=ax and y=ay then 940 else 460
840 rem ###### Aktionspunkt #########
850 if pixel(ax,ay)=0 then 880
860 color 1,0,1:linef ax*20-9,ay*20-
9,ax*20+9,ay*20+9
870 linef ax*20-9,ay*20+9,ax*20+9,ay*20-
9:goto 900
880 color 1,0,0:linef ax*20-9,ay*20-
9,ax*20+9,ay*20+9
890 linef ax*20-9,ay*20+9,ax*20+9,ay*20-
[illegible]
900 ax=x:ay=y
910 if pixel(ax,ay)=0 then 940
920 color 1,0,0:linef ax*20-9,ay*20-
9,ax*20+9,ay*20+9
930 linef ax*20-9,ay*20+9,ax*20+9,ay*20-
9:goto 460
```

Listing 1. »Der Maus-Editor« mit GEM-Funktionen

```
940 color 1,0,1:linef ax*20-9,ay*20-
9,ax*20+9,ay*20+9
950 linef ax*20-9,ay*20+9,ax*20+9,ay*20-
9
960 goto 460
970 rem ###### Cursor aktivieren #####
980 gosub auswertung
990 poke contrl,111:poke contrl+2,0:poke
contrl+6,37:poke intin,ax-1
1000  poke intin+2,ay-1:poke
intin+4,1:poke intin+6,0:poke intin+8,1
1010  for i=10 to 40 step 2
1020  poke intin+i,maskw(i/2-4):poke
intin+i+32,mausw(i/2-4):next
1030  vdisys
1040  x=18:y=7:color 1,1,1,0,0:gosub
rechteck:goto 460
1050  rem ##### abspeichern ########
1060  t$(1)="poke contrl+":t$(2)="poke
intin+"
1070  gosub auswertung
1080  i=12000
1090  open "O",#1,"cursor.bas",128
1100  ?#1,i;"data";:for y=1 to
7:?#1,maskw(y);",";:next:?#1,maskw(8):i=
i+10
1110  ?#1,i;"data";:for y=9 to 15:?#1,
maskw(y);",";:next:?#1,maskw(16):i=i+10
1120  ?#1,i;"data";:for y=1 to 7:?#1,
mausw(y);",";:next:?#1,mausw(8):i=i+10
1130  ?#1,i;"data";:for y=9 to 15:
?#1,mausw(y);",";:next:?#1,mausw(16):i=i
+10
1140  ?#1,i;t$(1);"0,111":i=i+10:?#1,i;
t$(1);"2,0":i=i+10:?#1,i;t$(1);"6,37"
1150  i=i+10:?#1,i;t$(2);"0,";ax-
1:i=i+10:?#1,i;t$(2);"2,";ay-1:i=i+10
1160  ?#1,i;t$(2);"4,1":i=i+10:?#1,i;
t$(2);"6,0":i=i+10:?#1,i;t$(2);"8,1"
1170  i=i+10:?#1,str$(i);"restore
12000":?#1,i;"for i=10 to 72 step
2":i=i+10
1180  ?#1,i;"read
wert:";t$(2);"i,wert":i=i+10:?#1,i;"next
":i=i+10
1190  ?#1,i;"vdisys"
1200  close #1
1210  x=18:y=9:color 1,1,1,0,0:gosub
rechteck
1220  goto 460
10000 rem ##### Maus - Position #######
10010 mauspos:
10020 poke contrl,124:poke contrl+2,0:
poke contrl+6,0:vdisys
10030 xmaus=peek(ptsout):ymaus=peek
(ptsout+2)-37:button=peek(intout):return
10040 rem ##### Mauscursor anschalten ##
10050 showcursor:
10060 poke contrl,122:poke contrl+2,0:
poke contrl+6,1:poke intin,0:vdisys
10070 return
10080 rem #### Mauscursor ausschalten #
10090 hidecursor:
10100 poke contrl,123:poke
contrl+2,0:poke contrl+6,0:vdisys
10110 return
10120 rem ####### Rechteck zeichnen ###
10130 rechteck:
10140 x1=x*20-s+1:x2=x*20+s+1:y1=y*20-
s+38:y2=y*20+s+38
10150 poke contrl,11:poke
contrl+2,2:poke contrl+6,0:poke
contrl+10,1
10160 poke ptsin,x1:poke ptsin+2,y1:poke
ptsin+4,x2:poke ptsin+6,y2:vdisys
10170 return
10180 rem ########### Text ausgeben ###
10190 text:
10200 poke contrl,11:poke contrl+2,2:
poke contrl+6,len(t$(z))+2:poke contrl
+10,10
10210 poke ptsin,x:poke ptsin+2,y:poke
ptsin+4,xdehn:poke intin,wdehn
10220 poke intin+2,zdehn
10230 for i=0 to len(t$(z))-1:poke
intin+i*2+4,asc(mid$(t$(z),i+1,1)):next
10240 vdisys
10250 return
10260 rem #### Schriftart ndern #######
10270 effekt:
10280 poke contrl,106:poke contrl+2,0:
poke contrl+6,1:poke intin,s:vdisys
10290 return
10300 rem #### Text-Gre ##############
10310 groesse:
10320 poke contrl,107:poke
contrl+2,0:poke contrl+6,1:poke
intin,hoehe:vdisys
10330 return
10340 rem #### Auswertung ############
10350 auswertung:
10360 for i=1 to 16:mausw(i)=0:
maskw(i)=0:next
10370 for y=1 to 16
10380 for x=16 to 1 step -1
10390 mausw(y)=mausw(y)+pixel(x,y)
*2^(16-x)
10400 maskw(y)=maskw(y)+maske(x,y)
*2^(16-x)
10410 next:next
10420 return
```

Listing 1. Der »Maus-Editor« mit GEM-Funktionen (Schluß)

```
12000 data 50048 , 42048 , 23200 , 11584
, 5824 , 2080 , 1104 , 1128
12010 data 1104 , 1312 , 1344 , 1344 ,
1344 , 1344 , 2336 , 3808
12020 data 0 , 17280 , 9536 , 4736 ,
2304 , 1984 , 928 , 912
12030 data 928 , 704 , 640 , 640 , 640 ,
640 , 1728 , 0
12040 poke contrl+0,111
12050 poke contrl+2,0
12060 poke contrl+6,37
12070 poke intin+0, 0
12080 poke intin+2, 0
12090 poke intin+4,1
12100 poke intin+6,0
12110 poke intin+8,1
12120 for i=10 to 72 step 2
12130 read wert:poke intin+i,wert
12140 next
12150 vdisys
```

Listing 2. So baut man seinen eigenen Maus-Cursor in Programme ein

J. Schultz/W. Pest
Drucker-Handbuch
Januar 1985, 188 Seiten

Welchen Drucker brauche ich? Einen schnellen, einen leisen, einen billigen, einen korrespondenzfähigen? Welcher Drucker kann problemlos an meinen Personal- oder Homecomputer angeschlossen werden? Wie funktioniert so ein Drucker überhaupt?
Lesen Sie dieses Buch, und Sie wissen auf alle diese Fragen eine fundierte Antwort!
• Endlich ein informativer Leitfaden für alle, die vor dem Kauf eines Druckers stehen.

Best.-Nr. MT 742
ISBN 3-89090-077-4 **DM 38,–**

J. Willis/M. Miller
Computertechnik ohne Geheimnisse
November 1984, 313 Seiten

Ob Sie schon einen Computer besitzen, seine Anschaffung planen oder sich nur mal aus Interesse mit diesem faszinierenden Gebiet auseinandersetzen wollen: in diesem Buch finden Sie die Antwort auf alle Fragen zur Computertechnik. Das Buch beschreibt den derzeitigen Stand der Technik, gibt Aufschluß über die Entwicklung der Computer und verschafft Ihnen eine Marktübersicht über Hard- und Software (wenn Sie nicht wissen, was das ist, dann lesen Sie dieses Buch!). Und es beantwortet auf amüsante Weise die oft gestellte Frage: »Was, zum Kuckuck, kann ich eigentlich mit so einem Ding machen?«.
• Informationen zum Thema Computer von allgemeinem Interesse.

Best.-Nr. MT 716
ISBN 3-89090-066-6 **DM 42,–**

Markt & Technik-Fachbücher erhalten Sie bei Ihrem Buchhändler.

Markt&Technik
BUCHVERLAG

Hans-Pinsel-Straße 2, 8013 Haar bei München

Depot-Händler

Tragen Sie Ihre Buchbestellung auf eine Postkarte ein und schicken diese an einen Depothändler in Ihrer Nähe oder an Ihren Buchhändler.

Buchhandlung Herder, Kurfürstendamm 69
1000 Berlin 15, Tel. (030) 8835002,
BTX *921782#
Computare Fachbuchhandlung, Keithstraße 18
1000 Berlin 30, Tel. (030) 2138021
Thalia Buchhaus, Große Bleichen 19
2000 Hamburg 36, Tel. (040) 3005050
Boysen + Maasch, Hermannstraße 31
2000 Hamburg 1, Tel. (040) 30050515
Electro-Data, Wilhelm-Heidsiek-Straße 1
2190 Cuxhaven, Tel. (04721) 51288
Buchhandlung Muehlau, Holtenauer Straße 116
2300 Kiel, Tel. (0431) 85085
ECL, Norderstraße 94-96
2390 Flensburg, Tel. (0461) 78181
Buchhandlung Weiland, Königstraße 79
2400 Lübeck, Tel. (0451) 74006-09
Buchhandlung Storm, Langenstraße 10
2800 Bremen 1, Tel. (0421) 321523
Buchhandlung Lohse-Eissing, Marktstraße 38
2940 Wilhelmshaven, Tel. (04421) 41887
Buchhandlung Schmorl u. v. Seefeld, Bahnhofstraße 13
3000 Hannover 1, Tel. (0511) 327651
Buchhandlung Graff, Neue Straße 23
3300 Braunschweig, Tel. (0531) 49271
Deuerlich'sche Buchhandlung, Weender Straße 33
3400 Göttingen, Tel. (0551) 56866
Buchhandlung an der Hochschule, Holländische Straße 22
3500 Kassel, Tel. (0561) 83807
Stern Verlag, Friedrichstraße 24-26
4000 Düsseldorf, Tel. (0211) 373033
Buchhandlung Baedeker, Kettwiger Straße 33-35
4300 Essen 1, Tel. (0201) 221381
Regensberg'sche Buchhandlung, Alter Steinweg 1
4400 Münster, Tel. (0251) 40541-5
Buchhandlung Achter, Johannisstraße 51
4500 Osnabrück, Tel. (0541) 28488
Buchhandlung Lensing, Westenhellweg 86-88
4600 Dortmund, Tel. (0231) 16980
Buchhandlung Brockmeyer, Querenburger Höhe 281/Unicenter
4630 Bochum, Tel. (0234) 701360
Buchhandlung Meier + Weber, Warburger Straße 98
4790 Paderborn, Tel. (05251) 63172
Buchhandlung Phönix GmbH, Obernitorwall 25
4800 Bielefeld 1, Tel. (0521) 69071
Buchhandlung Gonski, Neumarkt 24
5000 Köln 1, Tel. (0221) 210528
Mayer'sche Buchhandlung, Ursulinerstraße 17-19
5100 Aachen, Tel. (0241) 48142
Buchhandlung Behrendt, Am Hof 5a
5300 Bonn 1, Tel. (0228) 658021
Buchhandlung Cusanus, Schloßstraße 12
5400 Koblenz, Tel. (0261) 36239
Akad. Buchhandlung Interbook, Fleischstraße 61-65
5500 Trier, Tel. (0651) 43596
Buchhandlung W. Finke, Kipdorf 32
5600 Wuppertal 1, Tel. (0202) 454220
Buchhandlung Balogh, Sandstraße 1
5900 Siegen, Tel. (0271) 55298-9
Buchhandlung Naacher, Steinweg 3
6000 Frankfurt 1, Tel. (069) 298050
Buchhandlung Wellnitz, Lautenschlägerstraße 4
6100 Darmstadt, Tel. (06151) 76548
Buchhandlung Faller + Gecks, Friedrichstraße 31
6200 Wiesbaden, Tel. (06121) 304911
Ferber'sche Uni-Buchhandlung, Seltersweg 83
6300 Gießen, Tel. (0641) 12001
Sozialwissenschaftliche Fachbuchhandlung, Friedrichstraße 24
6400 Fulda, Tel. (0661) 75077
Gutenberg Buchhandlung, Große Bleiche 29
6500 Mainz, Tel. (06131) 37011
Buchhandlung Bock + Seip, Futterstraße 2
6600 Saarbrücken, Tel. (0681) 30677
Buchhandlung Wilhelm Hofmann, Bismarckstraße 98
6700 Ludwigshafen, Tel. (0621) 516001
Buchhandlung Loeffler, B 1,5
6800 Mannheim 1, Tel. (0621) 28912
Buchhandlung Stehn, Bahnhofstraße 13
7000 Stuttgart 60, Tel. (0711) 561476
Buchhandlung am Markt, Kramstraße 6
7100 Heilbronn, Tel. (07131) 68682
PCB Micro-Computer, Oskar-Kalbfell-Platz 8
7410 Reutlingen, Tel. (07121) 270443
UNI Buchhandlung Kellner + Moessner, Kaiserstraße 18
7500 Karlsruhe, Tel. (0721) 691436
Buchhandlung Roth, Hauptstraße 45
7600 Offenburg, Tel. (0781) 22097
Rombach Center, Bertholdstraße 10
7800 Freiburg, Tel. (0761) 49091
Fachbuchhandlung Hofmann, Hirschstraße 4
7900 Ulm, Tel. (0731) 60949
Schaufler Elektronik, Bachstraße 52
7980 Ravensburg, Tel. (0751) 26138
Buchhandlung Hugendubel, Marienplatz
8000 München 2, Tel. (089) 2389-1
Computerbücher am Obelisk, Barerstraße 32-34
8000 München 2, Tel. (089) 282383
Pete's Computerbücher, Schillerstraße 17
8000 München 2, Tel. (089) 555229
Universitätsbuchhandlung Lachner, Theresienstraße 43
8000 München 2, Tel. (089) 521340
Buchhandlung Schönhuber, Theresienstraße 6
8070 Ingolstadt, Tel. (0841) 33146/47
Computerstudio Gertrud Friedrich, Ludwigstraße 3
8220 Traunstein, Tel. (0861) 14767
Buchhandlung Pustet, Kl. Exerzierplatz 4
8390 Passau, Tel. (0851) 56945
Buchhandlung Pustet, Gesandtenstraße 6
8400 Regensburg, Tel. (0941) 53061
Buchhandlung Dr. Büttner, Adlerstraße 10-12
8500 Nürnberg, Tel. (0911) 232318
STS Computer Vertrieb, Werner-Siemens-Straße 19
8580 Bayreuth, Tel. (0921) 62320
Computer-Center-Burger, Leimitzer Straße 11-13
8670 Hof, Tel. (09281) 40075
Sortiments- u. Bahnhofsbuchh. J. Strykowski, Bahnhofplatz 4
8700 Würzburg, Tel. (0931) 54389
Buchhandlung Pustet, Grottenau 4
8900 Augsburg, Tel. (0821) 35437
Kemptener Fachsortiment, Salzstraße 30
8960 Kempten, Tel. (0831) 14413

Belgien:
Eicher Micro & Personal Computer, Hünningen 56-58
B-4780 St. Vith, Tel. (080) 227393

Luxemburg:
Librairie Promoculture, 14, rue Duchscher (Pl. de Paris)
L-1011 Luxembourg-Gare, Tel. 480691, Telex 3112

Schweiz:
Buchhandlung Meissner, Bahnhofstraße 41
5000 Aarau, Tel. (064) 247151
Bücher Balmer, Neugasse 12
6300 Zug, Tel. (042) 214141
Buchhandlung Enge, Bleicherweg 56
8002 Zürich, Tel. (01) 2012078
Buchhandlung Orell Füssli, Pelikanstraße 10
8022 Zürich, Tel. (01) 2118011
Freihofer AG, Wissenschaftliche Buchhandlung, Universitätstr. 11
8033 Zürich, Tel. (01) 3634282
Buchhandlung am Rösslitor, Webergasse 5
9001 St. Gallen, Tel. (071) 228726

Markt&Technik
BUCHVERLAG

Impressum

Herausgeber: Carl-Franz von Quadt, Otmar Weber

Chefredakteur: Michael Scharfenberger (sc)
Leitender Redakteur: Michael Lang (lg)
Redakteure: Horst Brandl (hb), Volker Everts (ev), Johannes Leckebusch (le), Heinrich Lenhardt (hl), Stefan Mutschler (sm), Markus Ohnesorg (og), Reiner Schönrock (rs), Boris Schneider (bs), Arnd Wängler (aw); Petra Wängler, Eva Hiermeier (Koordination)
Redaktionsassistenz: Monika Lewandowski (222)

Fotografie: Jens Jancke

Layout: Leo Eder (Ltg.), Sigrid Kowalewski (Cheflayouterin)

Auslandsrepräsentation:
Schweiz: Markt & Technik Vertriebs AG, Kollerstrasse 3, CH-6300 Zug, Tel. (042) 415656, Telex: 862329 mut ch
USA: M&T Publishing, 2464 Embarcadero Way, Palo Alto, CA 94303; Tel. 415-424-0600; Telex 752351

Manuskripteinsendungen: Manuskripte und Programmlistings werden gerne von der Redaktion angenommen. Sie müssen frei sein von Rechten Dritter. Sollten sie auch an anderer Stelle zur Veröffentlichung oder gewerblichen Nutzung angeboten worden sein, muß dies angegeben werden. Mit der Einsendung von Manuskripten und Listings gibt der Verfasser die Zustimmung zum Abdruck in von der Markt & Technik Verlags AG herausgegebenen Publikationen und zur Vervielfältigung der Programmlistings auf Datenträger. Mit der Einsendung von Bauanleitungen gibt der Einsender die Zustimmung zum Abdruck in von Markt & Technik Verlag AG verlegten Publikationen und dazu, daß Markt & Technik Verlag AG Geräte und Bauteile nach der Bauanleitung herstellen läßt und vertreibt oder durch Dritte vertreiben läßt. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte und Listings wird keine Haftung übernommen.

Produktionsleitung: Klaus Buck (180)

Anzeigenverkauf: Britta Fiebig (211)

Anzeigenverwaltung und Disposition: Patricia Schiede (172)

Marketingleiter Vertrieb: Hans Hörl (114)

Vertriebsleitung: Helmut Grünfeldt (189)

Verlagsleiter M&T Buchverlag: Günther Frank (212)

Vertrieb Handelsauflage: Inland (Groß-, Einzel- und Bahnhofsbuchhandel) sowie Österreich und Schweiz: Pegasus Buch- und Zeitschriften-Vertriebs GmbH, Hauptstätter Str. 96, 7000 Stuttgart 1, Tel. (0711) 6483-0

Bezugsmöglichkeiten: Leser-Service: Telefon (089) 4613-249. Bestellungen nimmt der Verlag oder jede Buchhandlung entgegen.

Bezugspreis: Das Einzelheft kostet DM 14,–.

Druck: Druckhaus München GmbH, Schellingstraße 39–43, 8000 München 40

Verantwortlich: Für redaktionellen Teil: Michael Scharfenberger
Für Anzeigen: Ralph Peter Rauchfuß (126)

Vorstand: Carl-Franz von Quadt, Otmar Weber

Anschrift für Verlag, Redaktion, Vertrieb, Anzeigenverwaltung und alle Verantwortlichen:
Markt & Technik Verlag Aktiengesellschaft,
Hans-Pinsel-Straße 2, 8013 Haar bei München,
Telefon (089) 4613-0, Telex 5-22052

Telefon-Durchwahl im Verlag:

Wählen Sie direkt: Per Durchwahl erreichen Sie alle Abteilungen direkt. Sie wählen 089/4613 und dann die Nummer, die in Klammern hinter dem jeweiligen Namen angegeben ist.

Aktionäre, die mehr als 25% des Kapitals halten: Otmar Weber, Ingenieur, München; Carl-Franz von Quadt, Betriebswirt, München; Aufsichtsrat: Dr. Robert Dissmann (Vorsitzender), Karl-Heinz Fanselow, Eduard Heilmayr

COMPUTER-ZEITSCHRIFTEN

VON PROFIS FÜR PROFIS

COMPUTER PERSÖNLICH

Das aktuelle Fachmagazin für Personal-Computer.

- ★ **Wenn Sie jetzt den Schritt vom Heim-Computer zur professionellen Anwendung eines Personal Computers planen**
- ★ **Wenn Sie beruflich oder privat bereits einen Personal Computer benutzen**
- ★ **Wenn Sie regelmäßig Informationen über das aktuelle Produktangebot benötigen**
- ★ **Wenn Sie selbst programmieren**
- ★ **Wenn Sie professionelle Hard- und Softwaretests suchen**
- ★ **Wenn Sie Ihr eigenes System möglichst effizient einsetzen wollen**

dann ist »Computer persönlich«, das aktuelle Fachmagazin für Personal Computer, genau Ihre Zeitschrift.

Die konsequente Ausrichtung auf professionelle Anwendungen bietet Ihnen alle wichtigen Informationen.

Von Profis für Profis!

»Computer persönlich« gibt es alle 14 Tage neu bei Ihrem Zeitschriftenhändler oder im Computer-Fachgeschäft.

PC MAGAZIN

Einzige Wochenzeitung für Personal Computer im IBM-Standard.

Sie beschäftigen sich beruflich oder privat mit dem Einsatz und der Anwendung von Personal Computern?

Sie sind an aktuellen, professionellen Informationen über IBM-PCs, kompatible Systeme und deren professionellen Einsatz interessiert? Dann ist das PC Magazin genau auf Ihre persönlichen Bedürfnisse zugeschnitten.

Es wird von anerkannten und erfahrenen Fachjournalisten für professionelle Anwender und Fachleute geschrieben.

Es berichtet jede Woche ausschließlich über Computer im IBM-Standard und kompatible Systeme, über Hard- und Softwareneuheiten. Es bringt ausführliche Testberichte und gibt Ihnen wichtige Informationen über Netzwerke sowie die PC/Host-Verbindung.

Nur diese Spezialisierung ermöglicht eine gezielte Berichterstattung und bietet genügend Raum, um auf Anwenderprobleme spezifisch eingehen zu können.

Von Profis für Profis!

Und das jeden Mittwoch neu bei Ihrem Zeitschriftenhändler oder im Computer-Fachgeschäft.

GUTSCHEIN

für ein kostenloses Probeexemplar

Senden Sie mir die neueste Ausgabe der von mir angekreuzten Zeitschrift kostenlos als Probeexemplar:

☐ **COMPUTER PERSÖNLICH**
Wenn mir Computer persönlich zusagt und ich es regelmäßig weiterbeziehen möchte, brauche ich nichts zu tun: Ich erhalte Computer persönlich dann regelmäßig alle 14 Tage per Post frei Haus geliefert und bezahle pro Jahr nur DM 98,— Zustellung und Postgebühren übernimmt der Verlag.

☐ **PC-MAGAZIN**
Wenn mir das PC-Magazin zusagt und ich es regelmäßig weiterbeziehen möchte, brauche ich nichts zu tun: Ich erhalte mein PC-Magazin dann regelmäßig jede Woche per Post frei Haus geliefert und bezahle pro Jahr nur DM 155,— Zustellung und Postgebühren übernimmt der Verlag.

Vorname/Name

Straße PLZ/Ort

Datum 1. Unterschrift

Mir ist bekannt, daß ich diese Bestellung innerhalb von 8 Tagen bei der Bestelladresse widerrufen kann und bestätige dies durch meine zweite Unterschrift. Zur Wahrung der Frist genügt die rechtzeitige Absendung des Widerrufs.

Datum 2. Unterschrift

Gutschein ausfüllen, ausschneiden, auf Postkarte kleben und einsenden an: Markt & Technik Verlag Aktiengesellschaft, Vertrieb, Postfach 1304, 8013 Haar.

Superbücher zum ATARI ST

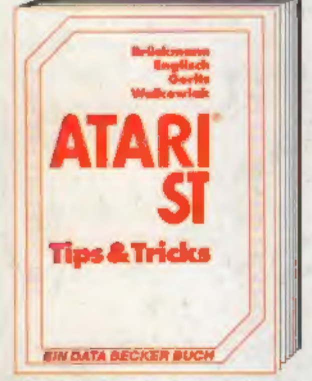
Der neue ATARI ist eine Supermaschine! Aber nur der richtige Einstieg garantiert den professionellen Umgang damit. Deshalb sollte dies Ihr erstes Buch sein. Eine Einführung in Handhabung, Einsatz und Programmierung des ST: die Tastatur, die Maus, der Editor, der erste Befehl, das erste Programm, der Anschluß der Geräte u.v.m. Dieses Buch ist ein Muß für jeden Einsteiger!
ATARI ST für Einsteiger, 262 Seiten, DM 29,–

Das Informationspaket zum ATARI ST mit ausführlicher Hardwarebeschreibung, detaillierter Erläuterung der Schnittstellen: V.24, Expansion-Interface, Aufbau und Funktionsweise der Maus, MIDI-Interface, Aufbau von Grafiken, BIOS, GEM, wichtige Systemadressen und was man damit machen kann. Unentbehrlich fürs professionelle Arbeiten mit dem ATARI ST.
ATARI ST INTERN, 464 Seiten, DM 69,–

Sie haben den Einstieg auf dem ATARI ST geschafft? Dann werden Sie mit diesem Buch zum Profi. Aus dem Inhalt: Datenfluß- und Programmablaufpläne, Grafik- und Soundprogrammierung, Sortierverfahren, Dateiverwaltung und viele nützliche Tips. Mit einer Befehlsübersicht incl. der nicht bekannten Befehle!
Das große BASIC-Buch zum ATARI ST, 268 Seiten, DM 39,–

LOGO ist keineswegs nur eine Sprache für Kinder, sondern eröffnet viele interessante Bereiche wie z.B.: Rechnen mit LOGO, Grafikprogrammierung, Wörter- und Listenverarbeitung, Prozeduren, Rekursionen, Sortierroutinen, Maskengenerator, Datenstrukturen und Künstliche Intelligenz. Mit LOGO können Sie schwierige und komplexe Probleme oft leichter lösen als mit anderen Programmiersprachen!
Das LOGO-Buch zum ATARI ST, ca. 300 Seiten, DM 49,–

Sie können BASIC und wollen „C" lernen? Mit diesem Buch kein Problem! Die elementaren Grundelemente wie Bildschirmoperationen, Variablen, Zeiger, arithmetische Ausdrücke und Kontrollstrukturen werden als Einführung benutzt, um weiterführende Sprachelemente wie Datenfelder, Strukturen und Funktionen zu erklären. So können Sie die Stärken von „C" schnell für eigene Programme ausnutzen!
Von BASIC zu C mit dem ATARI ST, 297 Seiten, DM 39,–



Eine riesige Fundgrube faszinierender Tips & Tricks um Ihren ATARI ST voll auszunutzen! Benutzung des ATARI-BASIC, Programmierung einer RAM-Disk, Druckerspooler und Farbhardcopies für Drucker und Plotter sind nur einige der umfangreichen Beispiele, die von DATA BECKER Spezialisten für Sie erstellt wurden. Ein fantastisches Buch zu einem fantastischen Rechner!
ATARI ST Tips & Tricks, 256 Seiten, DM 49,–

Schlagen Sie dem Betriebssystem Ihres ATARI ST ein Schnippchen. Wie? Mit PEEKS & POKES natürlich! Dieses Buch erklärt Ihnen den Umgang damit. Mit vielen wichtigen POKES und Ihren Anwendungsmöglichkeiten. Dabei wird der Aufbau Ihres ST's prima erklärt: Betriebssystem, Interpreter, Pointer und Stacks sind nur einige Stichworte dazu.
PEEKS & POKES zum ATARI ST, 194 Seiten, DM 29,–

Ein Buch für jeden, der unter GEM Programme erstellen will! Arbeiten mit der Maus, Icons, Virtual Device Interface, Application Environment System und Graphics Device Operating System. Ein besonderer Schwerpunkt liegt im Einbinden von GEM-Routinen in C und 68000-Assembler und der Programmierung in diesen Sprachen. GEM – das Betriebssystem der Zukunft!
Das große GEM-Buch zum ATARI ST, 459 Seiten, DM 49,–

Den ATARI ST voll ausnutzen können Sie nur in Maschinensprache! Zahlensysteme, Bitmanipulation, der 68000 im ATARI ST, Registerverwendung, Struktur des Befehlssatzes, Programmstrukturen, Rekursion, Stacks, Prozeduren, Grundlagen der Assemblerprogrammierung Schritt für Schritt, Verwendung von Systemroutinen und Tips zum Einbinden von Assemblerroutinen in Hochsprachen. Eine hervorragend geschriebene Einführung!
ATARI ST Maschinensprache, 250 Seiten, DM 39,–

Kein 68000-Programmierer sollte auf dieses Handbuch verzichten. Sie finden detailliertes Sachwissen zur Technik und Programmierung: Entwicklung des 68000, Aufbau, Signal- und Busbeschreibung, Peripheriebausteine, Befehlssatz, Programmbeispiele, Vergleich mit anderen 16-Bit-Prozessoren u.v.m. Ein Buch für echte Computerfreaks!
Das Prozessorbuch zum 68000, 516 Seiten, DM 59,–

Wichtig: Die DATA WELT 4/86 bringt den

ST SOFTWARE-FÜHRER

Alle Programme zum ATARI ST – unter diesem Motto steht ein ausführlicher Sonderteil der DATA WELT 4/86. Über 20 Seiten randvoll mit Kurztests, Tips & Tricks und Erfahrungsberichten. Außerdem wie in jeder DATA WELT viele weitere ST-Artikel und großer ST-Zubehörmarkt.

DATA WELT 4/86

ab 17. März am Kiosk

BESTELL-COUPON

Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1
Bitte senden Sie mir:

zzgl. DM 5,– Versandkosten
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name und Adresse bitte deutlich schreiben

DATA BECKER

Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010